NIST Special Publication 1500-101, Version 1.0

Election Event Logging Common Data Format Specification

Version 1.0

John Wack
Richard Rivello
Sam Dana
John Dziurlaj

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.1500-101

National Institute of Standards and Technology
U.S. Department of Commerce

NIST Special Publication 1500-101, Version 1.0

Election Event Logging Common Data Format Specification

Version 1.0

John Wack and Richard Rivello, *Software and Systems Division, Information Technology Laboratory, NIST*
Sam Dana, *Prometheus Computing*
John Dziurlaj, *Democracy Fund*

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.1500-103

April 2020

U. S. Department of Commerce
Wilbur Ross, Secretary

National Institute of Standards and Technology
Walter G. Copan, Under Secretary of Commerce for Standards and Technology and Director

National Institute of Standards and Technology (NIST) Special Publication 1500-101

31 pages (April 2020)

NIST Special Publication series 1500 is intended to capture external perspectives related to NIST standards, measurement, and testing-related efforts. These external perspectives can come from industry, academia, government, and others. These reports are intended to document external perspectives and do not necessarily represent official NIST positions.

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, federal agencies may wish to closely follow the development of these new publications by NIST.

Organizations are encouraged to review all draft publications during public comment periods and provide feedback to NIST. All NIST publications are available at http://www.nist.gov/publication-portal.cfm.

National Institute of Standards and Technology
Attn: Software and Systems Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8970) Gaithersburg, MD 20899-8930
Email: voting@nist.gov

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. This document reports on ITL's research, guidance, and outreach efforts in Information Technology and its collaborative activities with industry, government, and academic organizations.

Abstract

This publication describes an election event logging common data format specification for devices used in U.S. elections such as optical scanners, election management systems, and polling place devices. The data logged generally contains information about the conduct of the election, such as when the polls open, when a voter starts a voting session or casts a ballot, or when administrators logon to the devices, etc. The publication contains a UML model of the relevant election logging data and background information regarding requirements for election event logging in the Election Assistance Commission's Voluntary Voting System Guidelines.

Keywords

Common data format; elections; event; logging; timestamp; voting; VVSG.

Acknowledgements

The authors wish to thank their colleagues of the National Institute of Standards and Technology Voting Interoperability Public Working Group, who reviewed drafts of this document and contributed to its technical content. The editor gratefully acknowledges and appreciates the following contributors for their keen and insightful assistance with developing this specification:

Jim Cantor *Hart Intercivic*	McDermot Coutts *Unisyn*	Herb Deutsch *Election Systems and Software*
Joshua Franklin *The Turnout*	Arthur Keller *University of California*	James Long *Smartmatic*
Neal McBurnett *ElectionAudits*	John McCarthy *Verified Voting*	Ian Piper *Dominion Voting Systems*
Andrew Regenscheid *National Institute of Standards and Technology*	Paul Stenbjorn *Election Information Systems*	

In addition to the above acknowledgments, the authors also gratefully acknowledge and appreciate the National Institute of Standards and Technology's Mary Brady, James Foti, and Benjamin Long for their exceptional contributions in helping to improve the content of the publication. The authors also gratefully acknowledge and appreciate the significant contributions from individuals and organizations involved in the NIST Voting Interoperability Public Working Group as well as in the public and private sectors, whose thoughtful and constructive comments improved the overall quality, thoroughness, and usefulness of this publication.

Executive Summary

This document is a specification for a common data format (CDF) for the election-related logging information produced by election devices, including voting devices used in polling places such as scanners and ballot marking devices, and other voting equipment used to manage elections. The specification describes an election event logging UML (Unified Modeling Language) model and associated XML (eXtensible Markup Language) and JSON (JavaScript Object Notation) schemas that were generated from the model.

Election logs generally contain information relevant to the conduct of the election for which the election device is being used. This information includes important events such as when voting operations are enabled on the device, when a voter initiates a voting session, or when the device records that the voter has cast their ballot. Logs can include errors such as the inability of a device to record a vote due to an internal error or that the polls have been opened or closed prematurely multiple times during the election day. Election analysts can use this information to determine not only whether the device itself was performing correctly but also whether the device was used correctly in the election, that is, used accordingly to election procedures. Additionally, analysts can derive various statistics from the log files, such as how often voters arrived and initiated voting sessions or the amount of time on average it took to cast a ballot.

Currently, election devices do not create election logs in an interoperable common data format, but rather the log files are in proprietary formats and thus are more difficult for election analysts to read and analyze. If the election logging documentation is not at hand, the logs can be unintelligible or require extensive reverse engineering efforts. Thus, a common format for the election log information will make it significantly easier for election officials and analysts and testing labs to access and understand the log files and, potentially, make more informed use of the log files for purposes of election auditing, research, and testing.

This document contains discussion of the requirements in the Election Assistance Commission's (EAC) Voluntary Voting System Guidelines (VVSG) that specify the required and optional election event information to be logged. The publication also includes a second schema for manufacturers to document their specific event code used in their log files.

This specification is geared towards the following audiences:

- Election officials
- Voting equipment manufacturers
- Voting system testing laboratories
- Election-affiliated organizations
- The public

Table of Contents

List of Appendices

List of Figures

1 Introduction

This document is a specification for a common data format (CDF) for the election-related logging information produced by election devices, including voting devices used in polling places such as scanners and ballot marking devices, and other voting equipment used to manage elections. The specification describes an election event logging UML (Unified Modeling Language) [1] model and XML (eXtensible Markup Language) [2] and JSON (JavaScript Object Notation) [3] schemas that were generated from the model.

Election event logs contain information generated by voting-related applications such as for election management systems (EMS), electronic pollbook applications, or vote-capture applications that operate on the election devices. The sorts of information logged includes information required in the U.S. Election Assistance Commission (EAC) Voluntary Voting System Guidelines (VVSG) Version 1.1 [4], and it is expected that similar requirements for election event logs will be included in the next version of the VVSG currently under development. Manufacturers may also include additional information in the logs.

1.1 Why this Specification is Needed

The purpose of this specification is to provide a concise, interoperable XML and JSON formats for manufacturers to integrate into their voting equipment and for election offices, researchers, testing laboratories and other groups to use in their own software. Currently, manufacturers produce log files in proprietary formats, which are inconsistent across different manufacturers in the fields used for logging events. This makes analysis of log files more difficult, especially when multiple devices produced by multiple manufacturers are involved in the analysis. The advantages of using this specification include:

- Election logs are in the same, defined format regardless of device manufacturer;
- Manufacturers can use the same, defined format for defining event codes and other information that may be specific to their own equipment;
- Analyzing and testing election logs produced by different types of equipment and different manufacturers is made significantly easier.

1.2 Intended Audience

The intended audience of this specification includes election officials, manufacturers and developers, testing labs, as well as the public.

1.3 Document Structure

Section 2 starts with background and overview material on logging requirements in the EAC VVSG and how they are implemented in the UML model. Section 3 contains documentation for the UML schema. The appendices include references, definitions, acronyms, and file locations for downloading the schemas associated with this specification.

2 Background and Overview

This section provides background information about election equipment logging and requirements in the EAC VVSG that pertain to logging and are thus addressed by this specification. This section also shows how the requirements are implemented in the UML model.

Election applications such as EMS generally operate on devices including personal computers that themselves have an operating system and a logging capability. The VVSG 1.1 requires the election applications to generate logs of events that are deemed as significant, such as when the application allows a login by an administrator, or when the polls are opened and voting is enabled on the application, or when the application records a cast ballot (on an electronic vote capture device, for example). These events are generally written to a separate log file typically named the *election event log*.

2.1 VVSG logging requirements implemented

In Section 2.1.5.1 of the EAC's VVSG 1.1, Operational Requirements, Requirement D itemizes the basic data that voting equipment shall at a minimum log. Requirement D is as follows[1]:

> *The voting system equipment shall log at a minimum the following data characteristics for each type of event:*
>
> 1. *system ID;*
> 2. *unique event ID and/or type;*
> 3. *timestamp;*
> 4. *success or failure of event, if applicable;*
> 5. *User ID trigger* [sic] *the event, if applicable;*
> 6. *Resources requested, if applicable.*
>   - i. *Timekeeping mechanisms shall generate time and date values.*
>   - ii. *The precision of the timekeeping mechanism shall be able to distinguish and properly order all audit records.*
>   - iii. *Timestamps shall include the date and time, including hours, minutes and seconds.*
>   - iv. *Timestamps shall comply with ISO 8601 and provide all four digits of the year and include the applicable time zone.*
>   - v. *Voting system equipment shall only allow administrators to set or adjust the clock.*
>   - vi. *Voting system equipment shall limit clock drift to a minimum of one minute within a 15 hour period after the clock is set.*

The UML model in this specification implements the requirements within Requirement D and adds several additional optional fields for documentation purposes. Those systems that satisfy

[1] Clauses v and vi deal with access control to the time adjustment mechanism and capabilities to limit clock drift and are not addressed in this specification.

Requirement D can export directly into the format described by this specification or can include a translation capability to convert from the manufacturer format into this specification's format.

2.2 Use cases for this specification

There are three general use cases that this specification is intended to satisfy:

1. An election device creates an election event log and writes events to that log. The log may be on removable media or other memory.
2. An election device creates an election event log on removable media and writes events to that log; during election day the media may be removed and reinserted into a different election device, which continues to write events to the same log created by the first device.
3. Multiple election devices are connected to a logging device, which creates a single election event log and writes events from the multiple devices to that log.

The first use case results in the election device creating an event log and writing the device identification and other related information into the election event log, subsequently followed by the events recorded by the device. There is thus a one-to-one correspondence between the device and the election event log.

The second use case comes into play if a device being used in, say, a polling place, malfunctions and must be replaced. The election event log is being recorded on the device's removable media. When the removable media is inserted into the replacement device, the replacement device will continue to write to the same election event log file.

In the second use case, there is a many-to-one correspondence between the devices and the election event log. Thus, the election event log format is arranged such that the election events are properly associated with the corresponding generating devices.

The third use case is much like the second in that there will be multiple devices, however they will be connected to a logging device such as a server that creates an election event log and associates events with each of the connected devices. There is a one-to-one correspondence between the logging device and the election event log, and there is a many-to-one correspondence between the connected devices and the election event log. As with the second use case, the election event log format is arranged such that election events are properly associated with the corresponding generating devices.

2.3 UML Model

Figure 1 shows the UML model, which consists of 3 classes, one for describing information about the log file such as when generated, a second class for describing information about the device model, manufacturer, and other related information, and a third class to contain the logged details for individual events. The third class is associated with the second class so that

the election events are properly associated with the generating device. All 3 classes and their attributes correspond very closely to that of XML and JSON.

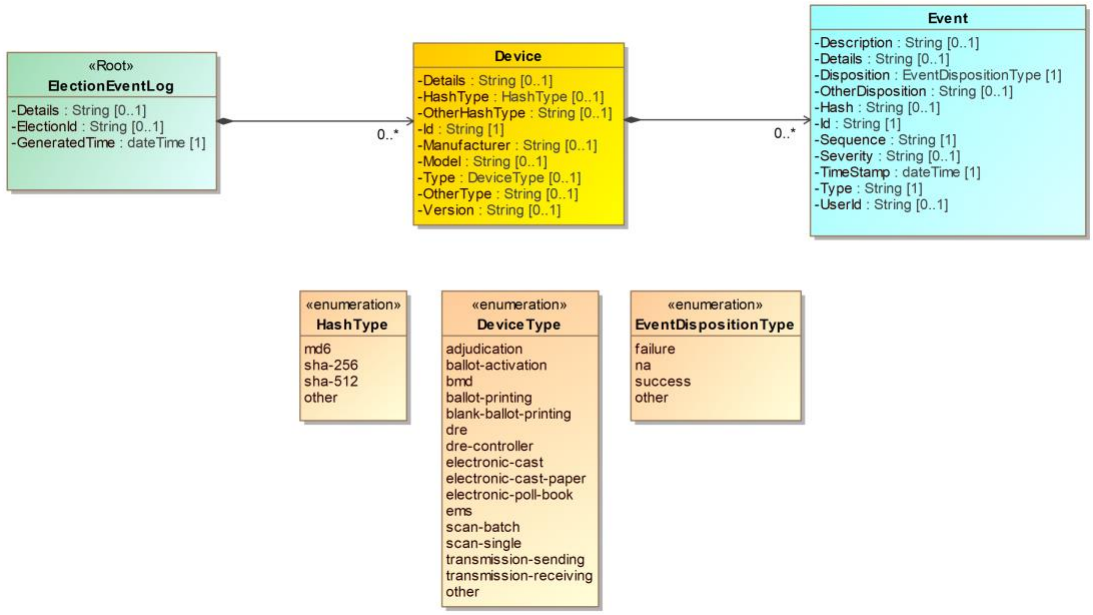

Figure 1 - Election Event Logging UML Class Diagram

2.3.1 UML Classes Overview

The ElectionEventLog class is the root class and contains information about the election event log file itself (as opposed to information about devices and election events). It includes:

- Details, for any details about the log file
- ElectionId, for identifying the election that the log file is specific to
- GeneratedTime, for the generation date/time of the election event log file

The ElectionEventLog class is the parent of one or more Device classes, which contain information about the device(s) generating the election events. Device includes:

- Details, for any details about the device that may need to be included
- HashType, for describing the type of cryptographic hash function used for event log entries
- Id, for identifying the device, for example by using an identifier such as a serial number
- Manufacturer, for identifying the name of the manufacturer
- Model, for identifying the model of the device
- Type, for describing the type of device

- Version, for identifying the version of the device

Each instance of the Device class is a parent of one or more ElectionEvent classes, which holds information about an election event. ElectionEvent includes:

- Description, a description of the event
- Details, for additional information the manufacturer may include
- Disposition, an indication of the event status
- Hash, for holding a cryptographic hash of the event
- Id, the identification number or string used by the manufacturer to identify the event
- Sequence, a unique identifier for the event, e.g., a sequential number
- Severity, an indication of severity, e.g., whether an event is critical, informational, etc.
- TimeStamp, when the event occurred
- Type, the type of event
- UserId, an identification of the user associated with or triggering the event

2.3.2 Examples of class associations to support use cases

Figures 2, 3, and 4 illustrate how the classes are associated to support the use general cases.

Figure 2 shows the associations for the first use case in which there is a single log file per device and multiple, unbounded election events associated with the device.

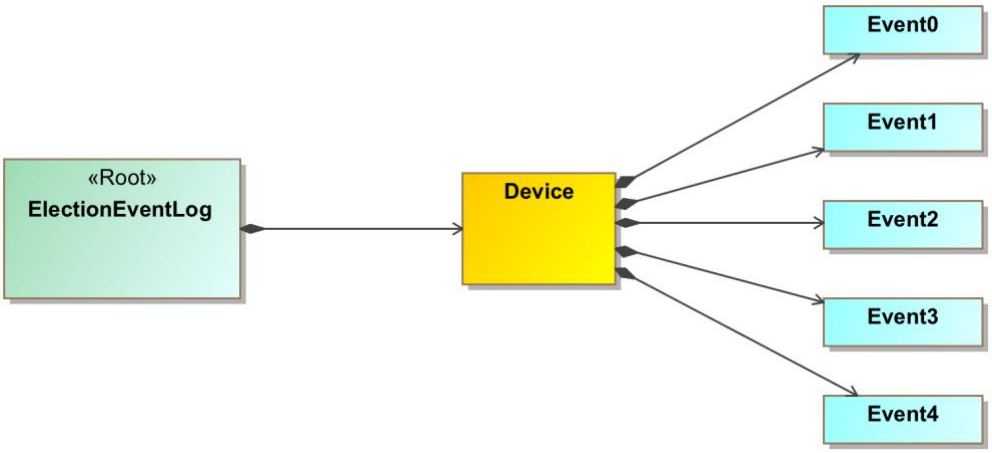

Figure 2 - First use case for a single log file per device

Figure 3 shows the associations for the second use case in which there may be multiple devices used successively to write to the same log file, which could occur if a device malfunctions, and

the removable media containing the log file is re-inserted into a replacement device.

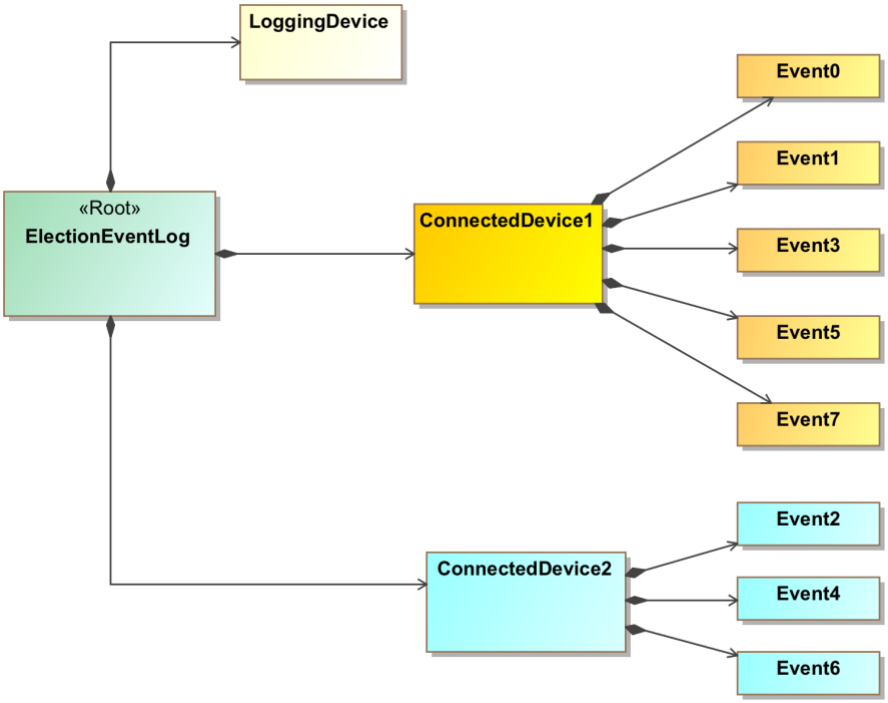

Figure 3 - Second use case for successive devices writing to same log file

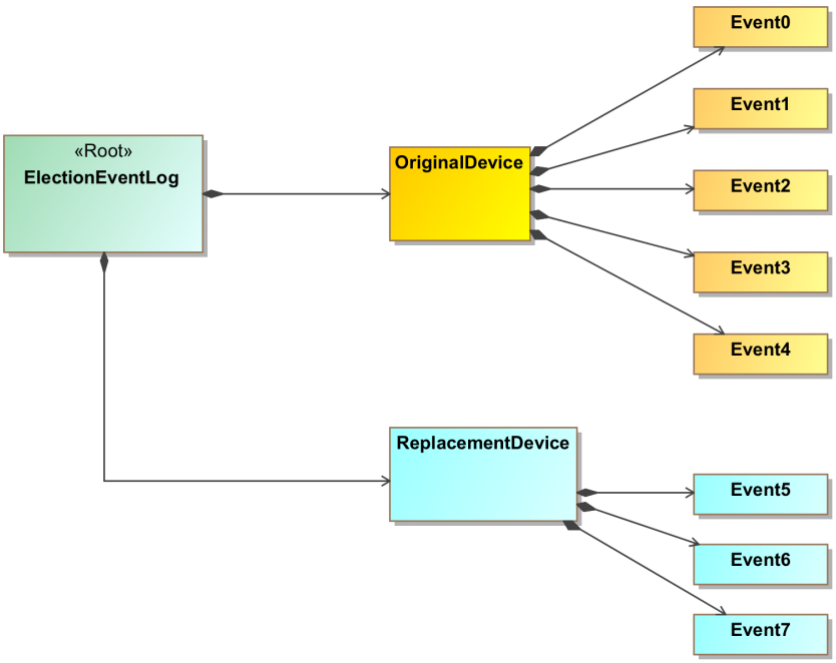

Figure 4 - Third use case for a logging device connected to event generating devices

Figure 4 shows the associations for the third use case in which a logging device is connected to multiple devices. The connected devices would send election events to the logging device, which writes events to the log file and maintains the correspondence between each device and its events. Figure 4 does not show the logging device itself generating any events, however this could happen in practice.

2.4 Documentation schema

To analyze an election log, one must understand the meaning of the various event IDs used in the log as well as, potentially, other fields. As configured today, manufacturer generally use their own specific event IDs, thus there is no consistency of the ID values or their meaning across different manufacturers.

One approach to making event IDs easier to understand when analyzing multiple manufacturer devices is to make them consistent in meaning and use. One could provide an event ID lexicon that would describe all known events and assign definitions/meanings to them, and each manufacturer could use this lexicon in all its election logs. Ultimately, however, this approach was not taken in this specification but could in a future version.

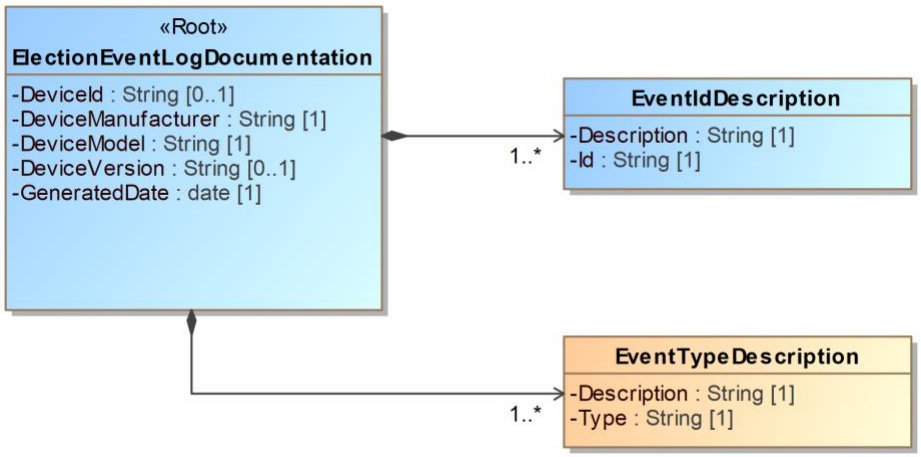

Figure 5 - Election Event Logging Documentation UML Class Diagram

The approach taken by this specification is to provide a second Election Event Logging Documentation UML model for containing the event ID documentation, along with documentation for each event type. Manufacturers can include a documentation file in this format for each of their devices. The model includes fields for device and manufacturer documentation, additional details, and then a series of fields for containing each event ID and its corresponding definition, and each event type and its corresponding definition.

3 Election Event Logging UML Model Documentation

This section contains documentation and discussion of the features included in the Election Event Logging and Election Event Logging Documentation UML models. As noted previously, this model was used in deriving the XML and JSON schemas, and the schema usage closely follows that of the UML model.

The UML classes are described first, followed by the enumerations. Each description contains an image of the class (from the UML model) and a table containing details about each of the class's attributes. To denote that certain class attributes derive from the class's associations with other classes, curly braces are used around those attribute names, e.g., if ClassA has an association with ClassB that is named "Automobile", then the table of attributes for ClassA would include "{Automobile}" as one of the attributes.

3.1 Classes

The following sections deal with the classes in the Election Event Logging and Election Event Logging Documentation UML models.

3.1.1 Class Device

Device contains information about the device generating election event logs. Id is the only required attribute, all other attributes are optional. If the device type is not found in the DeviceType enumeration, Type is 'other' and OtherType contains the appropriate type. Hash is used to specify a cryptograph hash associated with the events, that is, an event log entry, using values from the HashType enumeration. If the type of hash is not found in the HashType enumeration, HashType is 'other' and OtherHashType contains the type of hash.

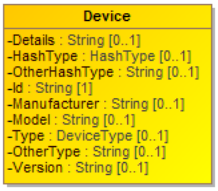

Figure 6 - Class Device

Attribute	Multiplicity	Type	Attribute Description
`Details`	0..1	`String`	Used to associate any details with the event log.
`{Event}`	0..*	`Event`	Used to describe a logged event.

Attribute	Multiplicity	Type	Attribute Description
HashType	0..1	String	The type of hash, from the HashType enumeration.
OtherHashType	0..1	String	If HashType is 'other', the type of the hash.
Id	1	String	A serial number or otherwise identifier associated with the device.
Manufacturer	0..1	String	Manufacturer of the device.
Model	0..1	String	Model of the device.
Type	0..1	DeviceType	Enumerated usage of the device, e.g., DRE, opscan-precinct, etc.
OtherType	0..1	String	Used when Type is ‘other’.
Version	0..1	String	Version identification of the device.

3.1.2 Class ElectionEventLog

ElectionEventLog is the root class of the Election Event Logging UML model. It includes Device for identifying the device(s) generating the election events, the date and time when the election event log was created, and an identification of the election. Details is used as needed for additional description/details.

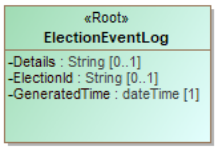

Figure 7 - Class ElectionEventLog

Attribute	Multiplicity	Type	Attribute Description
Details	0..1	String	Used to associate any details with the event log.
{Device}	0..*	Device	Used to describe the device(s) generating the election events.
ElectionId	0..1	String	Identifies the election associated with the log.
GeneratedTime	1	dateTime	Identifies the date and time the log was generated.

3.1.3 Class ElectionEventLogDocumentation

ElectionEventLogDocumentation is the root class of the Election Event Logging Documentation UML model. It includes EventIdDescription and EventTypeDescription, as well as other information for identifying the specific device associated with the election event documentation.

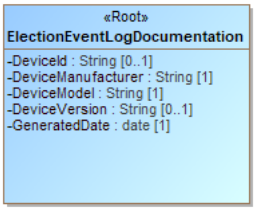

Figure 8 - Class ElectionEventLogDocumentation

Attribute	Multiplicity	Type	Attribute Description
DeviceId	0..1	String	A serial number or otherwise identifier associated with the device.
DeviceManufacturer	1	String	Manufacturer of the device.
DeviceModel	1	String	Model of the device.
DeviceVersion	0..1	String	Version identification of the device.
{EventIdDescription}	1..*	EventIdDescription	For associating a description with an event ID.
{EventTypeDescription}	1..*	EventTypeDescription	For associating a description with an event type.
GeneratedDate	1	date	Identifies the date the documentation report was generated.

3.1.4 Class Event

Event holds information about a specific event. Severity is an optional attribute for describing a severity indication for the event. If the event disposition is not found in the EventDispositionType enumeration, Disposition is 'other' and OtherDisposition contains the other disposition.

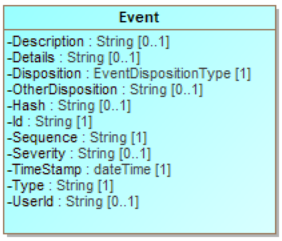

Figure 9 - Class Event

Attribute	Multiplicity	Type	Attribute Description
Description	0..1	String	Used for a brief description of the event.
Details	0..1	String	Used for additional information about the event, e.g., vendor reserved information.
Disposition	1	EventDispositionType	The disposition, e.g., success or failure, of the event.
OtherDisposition	0..1	String	Used when Disposition is 'other'.
Hash	0..1	String	Contains a cryptographic hash of the event [5], encoded as a string.
Id	1	String	An identifier associated with the event.
Sequence	1	String	A sequence number/string to uniquely identify the event in the log file.
Severity	0..1	String	Used for an indication of the severity of the event, as determined by the device vendor.
TimeStamp	1	dateTime	Identifies the date and time the event was generated.
Type	1	String	Used for the type of event, as

Attribute	Multiplicity	Type	Attribute Description
			determined by the device vendor.
UserId	0..1	String	An identifier associated with a user, as relevant.

3.1.5 Class EventIdDescription

For associating a brief description with an election event ID, used in ElectionEventLogDocumentation::EventIdDescription.

Figure 10 - Class ElectionEventId

Attribute	Multiplicity	Type	Attribute Description
`Description`	1	`String`	Used for a brief description of the event.
`Id`	1	`String`	An identifier associated with the event.

3.1.6 Class EventTypeDescription

For associating a description with an election event log type, used in ElectionEventLogDocumentation::EventTypeDescription.

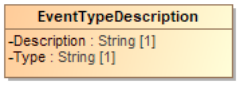

Figure 11 – Class EventTypeDescription

Attribute	Multiplicity	Type	Attribute Description
`Description`	1	`String`	Used for a description of the event type.
`Type`	1	`String`	An identifier associated with the event type.

3.2 Enumerations

The following section deal with the enumerations in the Election Event Logging and Election Event Logging Documentation UML models.

3.2.1 Enumeration DeviceType

Used in Device::Type to describe the type or usage of the device generating the event.

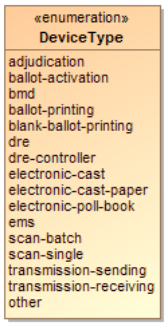

Figure 12 - Enumeration DeviceType

Value	Value Description
`adjudication`	Electronic adjudication function for reviewing absentee/mail-in ballots anomalies (blanks/overvotes/write-ins/unreadable ballots).
`ballot-activation`	Devices for enabling a vote capture device (VCD) to display a ballot, possibly directly connected to the VCD or through a smart card interface.
`bmd`	Ballot marking devices (voter facing).
`ballot-printing`	Marked ballot printing devices (voter facing).
`blank-ballot-printing`	On-demand blank ballot printers.
`dre`	Electronic voter stations, standalone or daisy chained to a DRE-controller (voter facing).
`dre-controller`	Network controller for electronic voting (poll worker facing)
`electronic-cast`	DREs, or other devices that store cast vote records electronically (voter facing).
`electronic-cast-paper`	DREs, or devices that store cast vote records electronically and also print a paper record (voter facing).
`electronic-poll-book`	Electronic poll book devices.

Value	Value Description
`ems`	Election management systems, including for pre- and post-election administration and reporting functions.
`scan-batch`	Scanning devices for batches of ballots, auto-feeding, e.g., Central Count (poll worker facing).
`scan-single`	Scanning devices for single-sheets, e.g., Precinct Count (voter facing), but could be used for Central Count by an election official.
`transmission-sending`	Remote transmission clients, e.g., for sending of unofficial results from a remote location to a central location (sending station).
`transmission-receiving`	Remote transmission hosts, e.g., for the receiving of unofficial results at a central location from a remote location (receiving station).
`other`	Used when no other value in this enumeration applies.

3.2.2 Enumeration EventDispositionType

Used in Event::Disposition for types of event dispositions.

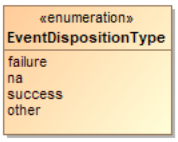

Figure 13 - Enumeration EventDisposition

Value	Value Description
`failure`	For a failure disposition.
`na`	Used when the disposition is not applicable or there is no disposition.
`success`	For a successful disposition.
`other`	Used when no other value in this enumeration applies.

3.2.3 Enumeration HashType

Used in Hash::Type to indicate the type of hash in use.

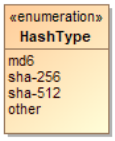

Figure 14 - Enumeration HashType

Value	Value Description
`md6`	To indicate that the MD6 message digest algorithm is being used.
`sha-256`	To indicate that the SHA 256-bit signature is being used.
`sha-512`	To indicate that the SHA 512-bit signature is being used.
`other`	Used when no other value in this enumeration applies.

Appendix A—Acronyms

Selected acronyms and abbreviations used in this document are defined below.

CDF	Common Data Format
DRE	Direct Record Electronic
EAC	Election Assistance Commission
EMS	Election Management System
ISO	International Standards Organization
JSON	JavaScript Object Notation
UML	Unified Modeling Language
VVSG	Voluntary Voting Systems Guidelines
XML	eXtensible Markup Language

Appendix B—Glossary

Selected terms used throughout this document are defined below.

Adjudication: Process of resolving flagged cast ballots to reflect voter intent. Common reasons for flagging include:

- write-ins,
- overvotes,
- marginal marks,
- having no contest selections marked on the entire ballot, or
- the ballot being unreadable by a scanner.

Cryptographic hash: A cryptographic algorithm that computes a numerical value based on a data file or electronic message. The numerical value is used to represent that file or message and depends on the entire contents of the file or message. A hash function can be considered to be a fingerprint of the file or message. Colloquially known as a hash, hash function, or digital fingerprint. Hashes provide integrity protection.

Digital signature: A cryptographic operation where the private key is used to digitally sign an electronic document and the public key is used to verify the signature. Digital signatures provide data authentication and integrity protection.

Direct record electronic (DRE): A vote-capture device that allows:

- electronic presentation of a ballot,
- electronic selection of valid contest options, and
- electronic storage of contest selections as individual records. It also provides a summary of these contest selections.

Election management system (EMS): Set of processing functions and databases within a voting system typically used to:

- develop and maintain election definition data,
- perform ballot layout functions,
- create ballot presentation templates for ballot printers or devices used by voters for ballot markup,
- count votes,
- consolidate and report results, and
- maintain audit trails.

Vote-capture device: An electronic voting device that is used directly by a voter to make selections on a ballot.

Appendix C—References

[1] Object Management Group (OMG), *UML Specification version 1.1* (OMG document ad/97-08-11) September 22, 2011, http://omg.org/ [accessed 02/09/2019].

[2] W3C, *Extensible Markup Language (XML) 1.0 (Fifth Edition)*, W3C Recommendation, November 26, 2008, http://www.w3.org/TR/xml/ [accessed 02/09/2019].

[3] JavaScript Object Notation, http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf [accessed 02/09/2019].

[4] Election Assistance Commission, *Voluntary Voting System Guidelines Version 1.1 Volume 1*, https://eac926.ae-admin.com/assets/1/Documents/VVSG.1.1.VOL.1.FINAL.pdf [accessed 2/9/2019].

[5] National Institute of Standards and Technology, *Special Publication 800-21 Second Edition, Guideline for Implementing Cryptography in the Federal Government*, December 2005, http://dx.doi.org/10.6028/NIST.SP.800-21e2 [accessed 2/9/2019].

Appendix D—File Download Locations

The files associated with this specification are available for download from a NIST repository, whose address is:

https://github.com/usnistgov/ElectionEventLogging

It can also be downloaded via NIST's voting program site:

http://vote.nist.gov

The files available to download include:

- This specification
- XML and JSON schemas
- UML models