Election Markup Language (EML) Specification Version 6.0

Public Review 02

18 June 2010

Specification URIs:
This Version:
http://docs.oasis-open.org/election/eml/v6.0/cd02/EML-Specification-v6.0-PR02.doc
http://docs.oasis-open.org/election/eml/v6.0/cd02/EML-Specification-v6.0-PR02.html
http://docs.oasis-open.org/election/eml/v6.0/cd02/EML-Specification-v6.0-PR02.pdf
(Authoritative)

Previous Version:
http://docs.oasis-open.org/election/eml/v6.0/cd01/EML-Specification-v6.0.doc
http://docs.oasis-open.org/election/eml/v6.0/cd01/EML-Specification-v6.0.html
http://docs.oasis-open.org/election/eml/v6.0/cd01/EML-Specification-v6.0.pdf (Authoritative)

Latest Version:
http://docs.oasis-open.org/election/eml/v6.0/EML-Specification-v6.0.doc
http://docs.oasis-open.org/election/eml/v6.0/EML-Specification-v6.0.html
http://docs.oasis-open.org/election/eml/v6.0/EML-Specification-v6.0.pdf

Technical Committee:
OASIS Election and Voter Services TC

Chair:
John Borras

Editors:
John Borras
David Webber

Related work:
This specification supercedes:
- Election Markup Language (EML) v5.0

See also:
- EML Data Dictionary
- EML Schemas
- EML Core Components

Declared XML Namespace:
urn:oasis:names:tc:evs:schema:eml

Abstract:
This document describes the background and purpose of the Election Markup Language, the electoral processes from which it derives its structure and the security and audit mechanisms it is designed to support. It also provides an explanation of the core schemas used throughout, definitions of the simple and complex datatypes, plus the EML schemas themselves. It also

covers the conventions used in the specification and the use of namespaces, as well as the guidance on the constraints, extendibility, and splitting of messages.

Status:

This document was last revised or approved by the Election and Voter Services Technical Committee on the dates shown in Appendix C – Revision History. The level of approval is also listed above. Check the "Latest Version" or "Latest Approved Version" location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/election/

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/election/ipr.php

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/election/.

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

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

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

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

1. Executive Summary

OASIS, the XML interoperability consortium, formed the Election and Voter Services Technical Committee in the spring of 2001 to develop standards for election and voter services information using XML. The committee's mission statement is, in part, to:

"Develop a standard for the structured interchange among hardware, software, and service providers who engage in any aspect of providing election or voter services to public or private organizations..."

The original objective in 2001 was to introduce a uniform and reliable way to allow systems involved in the election process to interact. The overall focus today provides a rich standard that is:

- **Multinational:** Our focus is to have standards that can be adopted globally.
- **Flexible**: Effective across the different voting regimes (e.g. proportional representation or 'first past the post') and voting channels (e.g. Internet, SMS, postal or traditional paper ballot).
- **Multilingual:** Flexible enough to accommodate the various languages and dialects and vocabularies.
- **Adaptable:** Resilient enough to support elections in both the private and public sectors.
- **Secure:** Able to secure the relevant data and interfaces from any attempt at corruption, as appropriate to the different requirements of varying election rules.
- **Technology agnostic:** technologically stable and forward deployable with backward feature compatibility

The primary deliverable of the committee is the Election Markup Language (EML). This is a set of data and message definitions described as XML schemas along with a dictionary of core terms and structures that enable predictable and consistent foundation mechanisms. The messages that form EML are intended for transfer between systems. It is not intended that all aspects of an election system will have a corresponding schema.

At present EML includes specifications for:

- Candidate Nomination, Response to Nomination and Approved Candidate Lists
- Referendum Options Nomination, Response to Nomination and Approved Options Lists
- Voter Registration information, including eligible voter lists
- Various communications between voters and election officials, such as polling information, election notices, district boundaries, polling places, facilities and services provided, eligibility, etc.
- Ballot information (races, contests, issues, candidates, etc.)
- Voter Authentication
- Vote Casting and Vote Confirmation
- Election counts, statistics and results
- Audit information pertinent to some of the other defined data and interfaces
- EML is flexible enough to be used for elections and referendums that are primarily paper-based or that are fully e-enabled.

This document and its accompanying set of schemas do not claim to satisfy the final requirements of any and all registration or election systems. The specification represents our best current efforts, knowledge and experience with election systems since 2001. It is incumbent on the users of this document to identify any requirement gaps, mistakes, inconsistencies or missing data and to propose corrections or enhancements to the OASIS Election and Voter Services Technical Committee.

1.1 Overview of the Document

To help establish context for the specifics contained in the XML schemas that make up EML, the committee also developed a generic end-to-end election process model. This model identifies the significant components and processes common to many elections and election systems, and describes how EML can be used to standardize the information exchanged between those components.

Section 2 outlines the business and technical needs the committee is attempting to meet, the challenges and scope of the effort, and introduces some of the key framing concepts and terminology used in the remainder of the document.

Section 3 describes two complementary high-level process models of an election exercise, based on the human and technical views of the processes involved. It is intended to identify all the generic steps involved in the process and highlight all the areas where standardized data is to be exchanged or referenced. The discussions in this section presents details of how the messages and data formats detailed in the EML specifications themselves can be used to achieve the goals of open interoperability between system components. Also contained in this Section are high-level data models showing the relationships of the data used in the election processes.

Section 4 provides an overview of the approach that has been taken to creating the XML schemas.

Section 5 provides descriptions of the core elements, data types and schemas developed to date.

Appendices provide information on internet voting security concerns; use of the EML defined TimeStamp schema; the W3C Digital Signature technology; and Acknowledgements and a revision history.

1.2 Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in **[RFC2119]**.

1.3 Normative References

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

1.4 Non-Normative References

[xNAL] OASIS Committee Specification 03 “eXtensible Name and Address (xNAL) Specifications and Description Document Version 3.0” October 2008 http://docs.oasis-open.org/ciq/v3.0/specs/ciq-specs-v3.pdf

[UK’s APD] Address and Personal Details Fragment v1.1 Technology Policy Team, e-Government Unit, Cabinet Office UK, 1 March 2002 http://www.govtalk.gov.uk/interoperability/draftschema_schema.asp?schemaid=92

[XML] Extensible Markup Language (XML) 1.0 (Third Edition) Tim Bray et al, Worldwide Web Consortium, 4 February 2004 http://www.w3.org/TR/REC-xml

[XML-DSig] XML-Signature Syntax and Processing Donald Eastlake et al, Worldwide Web Consortium, 10 June 2008 http://www.w3.org/TR/xmldsig-core/

[VoiceXML] Voice Extensible Markup Language (VoiceXML) Version 2.0 Scott McGlashan et al Worldwide Web Consortium 16 March 2004 http://www.w3.org/TR/voicexml20

2. Introduction

2.1. Business Drivers

Voting is one of the most critical features in our democratic process. In addition to providing for the orderly transfer of power, it also cements the citizen's trust and confidence in an organization or government when it operates efficiently. In the past, changes in the election process have proceeded deliberately and judiciously, often entailing lengthy debates over even the minutest detail. These changes have been approached with caution because discrepancies with the election system threaten the very principles that make our society democratic.

Society has become network oriented and citizens, used to the high degree of flexibility in the services provided by the private sector and in the Internet in particular, are now beginning to set demanding standards for the delivery of services by governments using modern electronic information systems.

The implementation of electronic voting (e-voting) has become globally widespread allowing increased access to information in the voting process for citizens everywhere and offering the scope for better verification and oversight for election supervision procedures. Allowing better access to information with consistent transparency and verification of results across the whole election process helps foster greater engagement and participation of voters throughout the whole democratic process itself. This also requires that standards ensure that the process is clear, robust and precisely understood so that confidence in the results is ensured. Access to a standard process also allows solution vendors to participate in an open marketplace that stimulates cost effective delivery and adoption of new technology without obsolescing existing investments.

However, it is recognized that more traditional verification methods and oversight will continue to be vital and in fact more so with the use of technology. Strong democracy requires participation from citizens and continuous independent monitoring of processes, procedures and outcomes. The OASIS EML standard seeks to facilitate precisely that transparency, access and involvement for citizens to the election process, end to end.

2.2. Technical Drivers

In the election industry today, there are a number of different service vendors around the world, all integrating different levels of automation, operating on different hardware platforms and employing different solution architectures. With the global focus on e-voting systems and initiatives, the need for a consistent, auditable, automated and interoperable election system has never been greater.

The introduction of end-to-end open standards for election solutions is intended to enable election officials around the world to build upon existing infrastructure investments to evolve their systems as new technologies emerge. This will simplify the election process in a way that was never possible before. Open election standards as such aim to instill confidence in the democratic process among citizens and government leaders alike, particularly within emerging democracies where the responsible implementation of the new technology is critical.

2.3. The E&VS Committee

OASIS, the XML interoperability consortium, formed the Election and Voter Services Technical Committee to standardize election and voter services information using XML. The committee is focused on delivering and maintaining a **reliable, accurate and trusted** XML specification (Election Markup Language (EML)) for the structured interchange of data and referencing of data among hardware, software and service vendors who provide election systems and services.

EML is the leading XML specification of its kind. When implemented, it can provide a uniform, secure and verifiable way to allow e-voting systems to interact as global election processes evolve and are adopted.

The Committee's mission statement is:

"To develop a standard for the structured interchange of data among hardware, software, and service providers who engage in any aspect of providing election or voter services, be they partly paper-based or fully e-enabled, to public or private organizations. The services performed for such elections and referenda include but are not limited to:

- *candidate nomination,*
- *referendum options nomination,*
- *voter registration,*
- *polling places, districting and boundaries*
- *various communications between voters and elections officials,*
- *ballot information*
- *voter authentication*
- *vote casting and vote confirmation*
- *election counts, statistics and results.*"

The primary function of an electronic voting system is to capture voter preferences reliably, securely and report them accurately with legally requirements for privacy met correctly. Capture is a function that occurs between 'a voter' (individual person) and 'an e-voting system' (machine). It is critical that any election system be able to prove that a voter's choice is captured correctly and anonymously, and that the vote is not subject to tampering, manipulation or other frauds.

These universal democratic principles[1] can be summarized as a list of fundamental requirements, or 'six commandments', for electronic voting systems:

1 Keep each voter's choice an inviolable secret.

2 Allow each eligible voter to vote only once, and only for those offices for which he/she is authorized to cast a vote.

3 Do not permit tampering with the voting systems operations, nor allow voters to sell their votes.

[1] First developed by Dr. Michael Ian Shamos, a PhD Researcher who worked on 50 different voting systems since 1980 and who reviewed the election statutes in half the US states, along with review from other researchers on e-voting principles.

4 Report all votes accurately

5 The voting system shall remain operable throughout each election.

6 Keep an audit trail to detect any breach of [2] and [4] but without violating [1].

In addition to these business and technical requirements, the committee was faced with the additional challenges of specifying a requirement that was:

- Multinational – our focus is to have these standards adopted globally
- Effective across the different voting regimes – for example, proportional representation or 'first past the post', preferential voting, additional member system
- Multilingual – our standards will need to be flexible enough to accommodate the various languages and dialects and vocabularies
- Adaptable – our aim is to provide a specification that is resilient enough to support elections in both the private and public sectors
- Secure – the standards must provide security that protects election data and detects any attempt to corrupt it.

The Committee has followed these guidelines and operated under the general premise that any data exchange standards must be evaluated with constant reference to the public trust.

2.4. Challenge and Scope

The goal of the committee has been to develop an Election Markup Language (EML) for end-to-end use within the election process. This is a set of data and message definitions described as a set of XML schemas and covering a wide range of transactions that occurs end-to-end during various phases and stages of the life cycle of an election. To achieve this, the committee decided that it required a common terminology and definition of election processes that could be understood internationally. The committee therefore started by defining the generic election process models described here.

These processes are illustrative, covering the vast majority of election types and forming a basis for defining the Election Markup Language itself. EML has been designed such that elections that do not follow this process model should still be able to use EML as a basis for the exchange of election-related messages.

EML is focused on defining open, secure, standardized and interoperable interfaces between components of election systems and thereby providing transparent and secure interfaces between various parts of an election system. The scope of election security, integrity and audit included in these interface descriptions and the related discussions are intended to cover security issues pertinent only to the standardized interfaces and not to the internal or external security requirements of the various components of election systems.

The security requirement for the election system design, implementation or evaluation must be placed within the context of the vulnerabilities and threats analysis of a particular election scenario. As such the references to security within EML are not to be taken as comprehensive requirements for all election systems in all election scenarios, nor as recommendations of sufficiency of approach when addressing all the security aspects of election system design, implementation or evaluation. In fact, the data security mechanisms described in this document are all optional, enabling compliance with EML without regard for system security at all. It is anticipated that implementers may develop a complementary document for a specific election scenario, which refines the security issues defined in this document and determines their specific strategy and approach by leveraging what EML provides.

EML is meant to assist and enable the election process and does not require any changes to traditional methods of conducting elections. The extensibility of EML makes it possible to adjust to various e-democracy processes without affecting the process. Conceptually EML simply enables the exchange of data between the various end-to-end election stages and processes in a standardized way.

The solution outlined in this document is non-proprietary and will work as a template for any election scenario using electronic systems for all or part of the process. The objective is to introduce a uniform and reliable way to allow election systems to interact with each other. The OASIS EML standard is intended to reinforce public confidence in the election process and to facilitate the job of democracy builders by introducing guidelines for the selection or evaluation of future election systems.

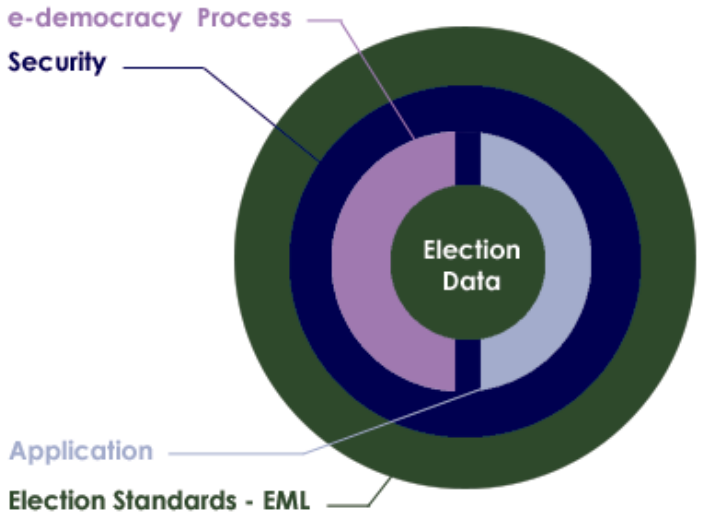


Figure 1A: e-Voting Components Relationship Overview

2.5. Documentation Set

To meet our objectives, the committee has defined a process model that reflects the generic processes for running elections in a number of different international jurisdictions. The processes are illustrative, covering a large number of election types and scenarios.

The next step was then to isolate all the individual data items that are required to make each of these processes function. From this point, our approach has been to use EML as a simple and standard way of exchanging this data across different electronic platforms. Elections that do not follow the process model can still use EML as a basis for the exchange of election-related messages at interface points that are more appropriate to their specific election processes. The EML standard is being used in a number of situations across a number of different international jurisdictions.

The document set comprises:

- **Specification:** This document. A general and global study of the electoral process. This introduces the transition from a complete manual election management process to a digitally enabled end-to-end election system by defining the data structures of content to be exchanged and or produced and where these data structures are needed, and describe how those exchanges and artifacts are encoded as XML schemas.
- **Data Requirements:** A data dictionary defining the data used in the processes and required to be handled by the XML schemas. The data dictionary is provided in both XML and spreadsheet formats. In addition there are data models available in the 'EML v6.0 Data Models' file.
- **EML Schemas:** This consists of a library of the XML schemas used in EML. The XML schemas define the formal structures of the election data that needs to be processed throughout an election.

- **EML Core Components Dictionary:** A dictionary containing full definitions of the elements and data types used by the EML Core schema. The core dictionary is provided in both XML and spreadsheet formats.
- **Templates:** for each schema a template is provided that facilitates generation of localizations of the main schema structure, creation of test case examples and implementation documentation. This aims to reduce implementer's costs of development and integration.

2.6. Voting Terminology

At the outset of our work, it was clear that the committee would need to rationalize the different terms that are commonly used to describe the election process.

Terms used to describe the election process, such as ballot and candidate, carry different meanings in different countries – even those speaking the same language. In order to develop a universal standard, it is essential to create universal definitions for the different elements of the election process. See the Data Dictionary for the terms used by the committee in this document

Our approach was to regard elections as involving Contests between Candidates or Referendum Options which aggregate to give results in different Elections.

In practice however, electoral authorities would often run a number of different elections during a defined time period. This phenomenon is captured in our terminology as an Election Event. Figure 1B uses a national parliamentary election process context to describe our approach in general terms.

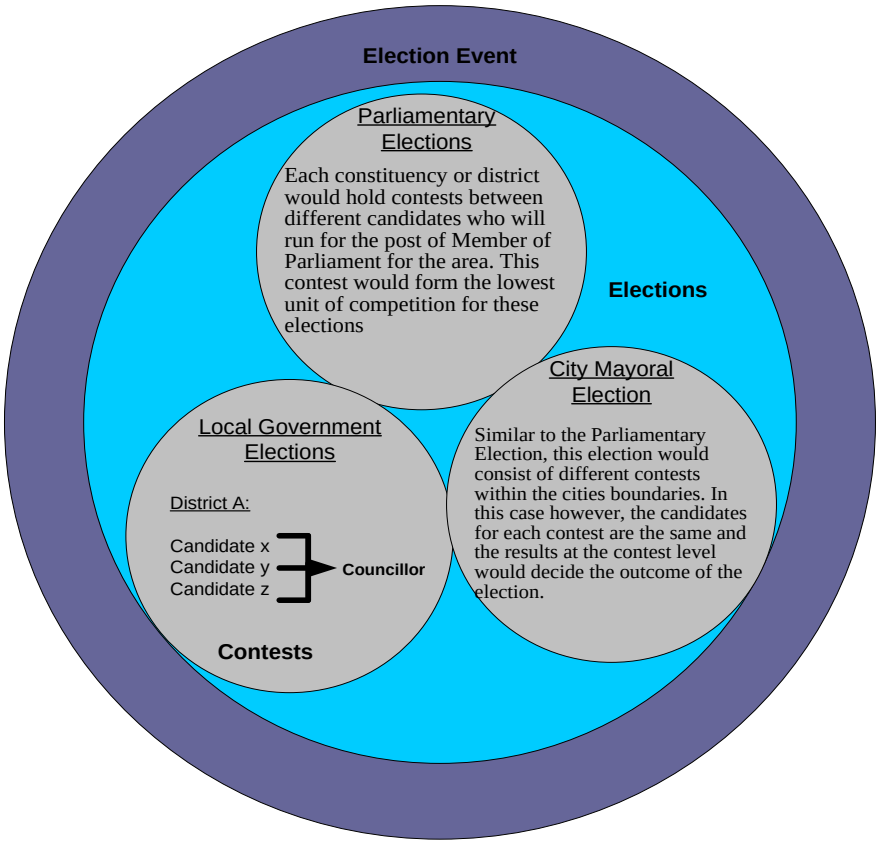


Figure 12B: The Election Hierarchy

In Figure 1C, there is an Election Event called the ‘Union Annual Election’. This comprises two Elections, one for the National Executive Committee (NEC) and one for the International Liaison Committee (ILC). Three positions are being selected for each committee; as a result, each Election is made up of three Contests. In region 1 (R1), the Contest for each Election has two Candidates.

Figure 1C shows the three Ballots (one for each region). The Ballot is personal to the voter and presents the Candidates available to that voter. It also allows choices to be made. During the election exercise, each voter in region 1 (R1) receives only the region 1 ballot. This ballot will contain the Candidates for the R1 contest for each of the two Elections.

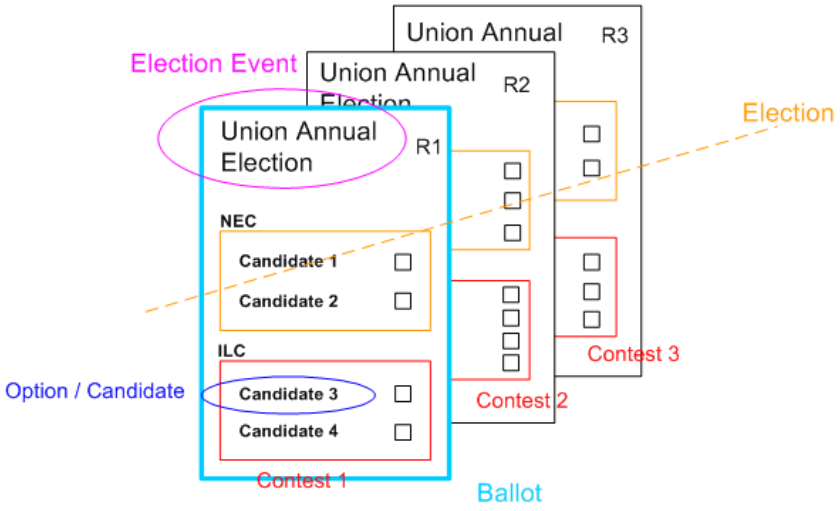


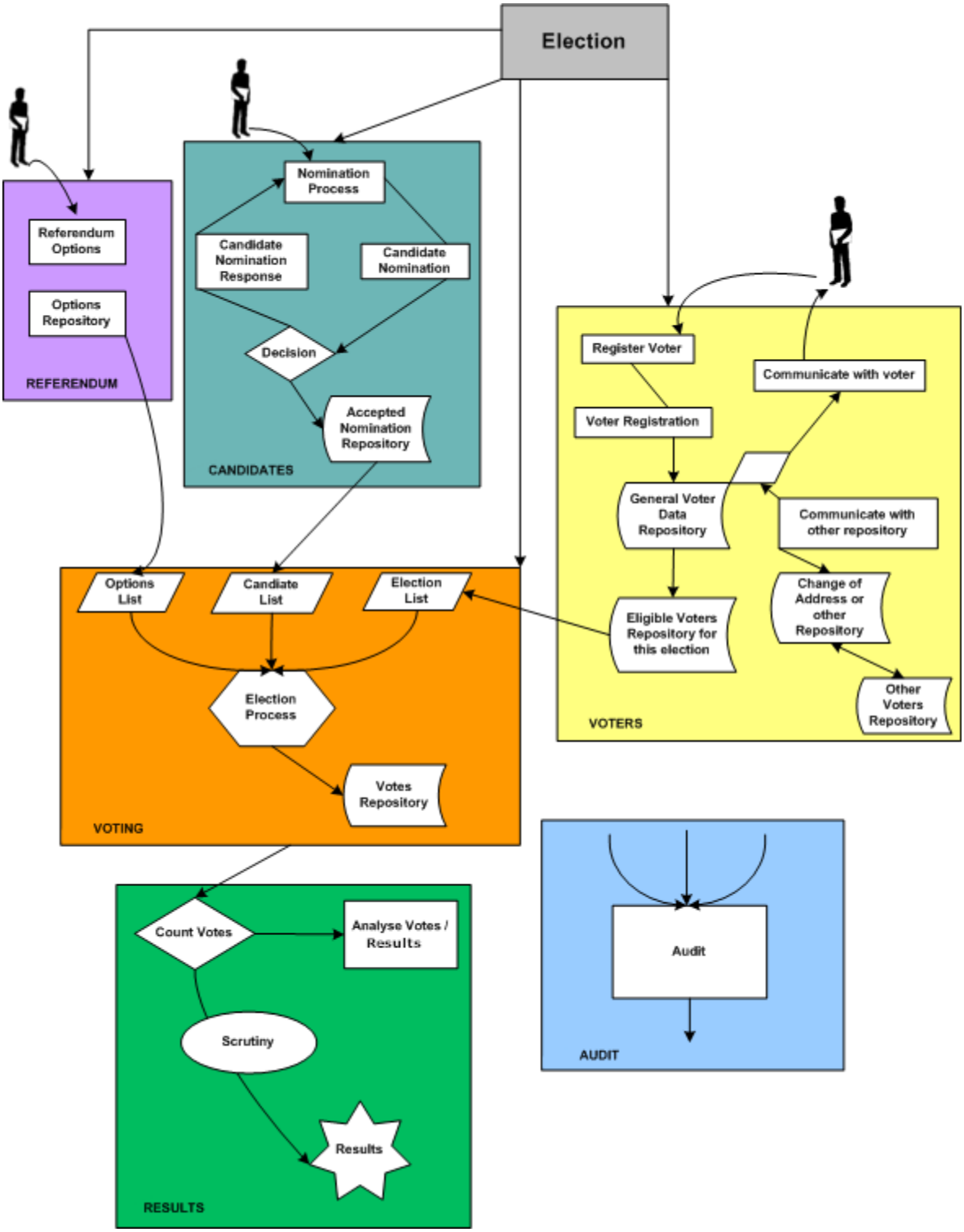
Figure 13C: Union Annual Election Event

3. High-Level Election Process

Section 3 describes two complementary high level process models of an election exercise, based on the human and technical views of the processes involved. It is intended to identify all the generic steps involved in the process and highlight all the areas where data is to be exchanged.

First two diagrams are presented (Figures 2a and 2B below) that illustrate these process models and then the section continues by providing details pertaining to the models and illustrative real world processes they introduce.

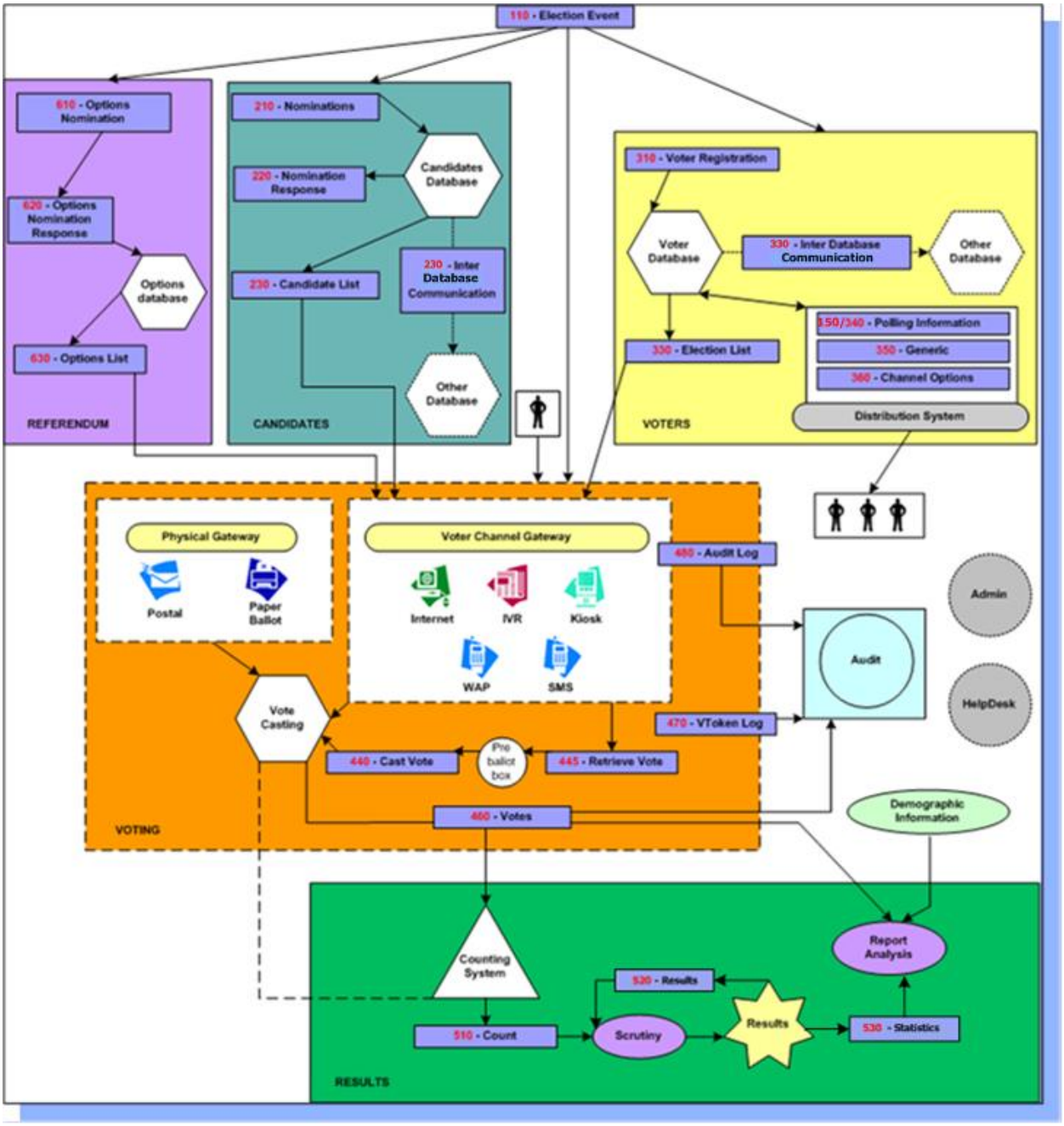
Figure 2A: High Level Model – Human View



275 *Figure 2B: High Level Model – Technical View*

276

277



278

279

280

3.1. Outline

This *high-level process model* is derived from real world election experience and is incorporates knowledge gained over the past 8 years of refining and improving the specification for EML.

For clarity, the whole process can be divided into 3 major areas, pre election, election, post election; each area involves one or more election processes. This document allocates a range of numbers for each process. One or more XML schemas are specified to support each process, this ensures consistency with all the figures and the schemas required:

- Pre election
  - Election (100)
  - Candidates (200)
  - Options (600)
  - Voters (300)
- Election
  - Voting (400)
- Post election
  - Results (500)
  - Audit
  - Analysis

Some functions belong to the whole process and not to a specific part:

- Administration Interface
- Help Desk

3.2. Process Descriptions

3.2.1. The Candidate Nomination Process

This is the process of approving nominees as eligible candidates for certain positions in an election. A candidate in this context can be a named individual or a party.

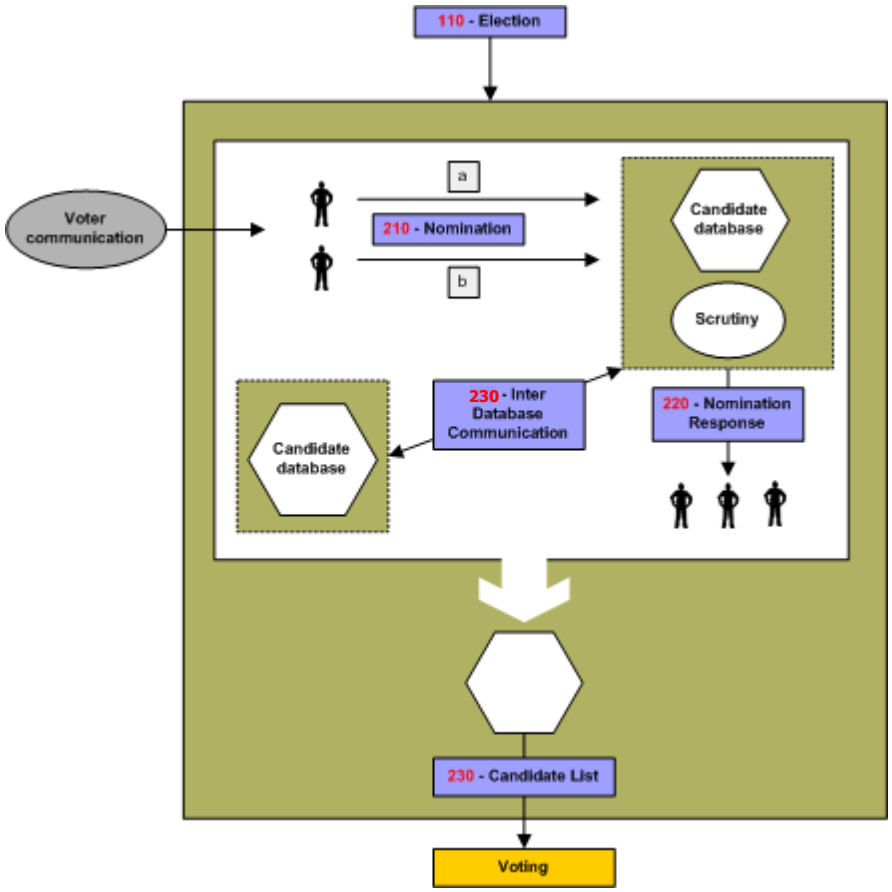


Figure 2C: The Candidate Nomination Process

Irrespective of local regulations covering the nomination process, or the form in which a candidate's nomination is to be presented, (e.g. written or verbal), the committee anticipates that the process will conform to the following format:

- Voter Communications [350-Generic] declaring the opening of nominations will be used to reach the population eligible to nominate candidates for a position x in an election y.
- Interested parties will respond in the proper way satisfying the rules of nomination for this election with the objective of becoming running candidates. The response message conforms to schema 210.
- A nomination for an individual candidate can be achieved in one of two ways:
- A Nominee will reply by attaching to his nomination a list of x number of endorsers with their signature.
- Each endorser will send a message specifying Mr. X as his or her nominee for the position in question. Mr X will signal his agreement to stand.

Note that nomination and the candidate's agreement to stand might be combined in a single message or sent as two messages, each conforming to schema 210.

The election officer(s) of this specific election will scrutinize those replies by making sure the requirements are fully met. Requirements for nomination vary from one election type to another, for example some elections require the nominee to:

- Pay fees,

- Have x number of endorsers,
- Be of a certain age,
- Be a citizen more than x number of years,
- Not stand for election in more than one contest at a time,
- Etc.

Schema 210 provides mechanisms to identify and convey scrutiny data but since the laws of nomination vary extensively between election scenarios, no specific scrutiny data is enumerated.

Schema 330 allows election officials to enquire of other jurisdictions whether a particular candidate is standing in more than one contest.

Nominees will be notified of the result of the scrutiny using a message conforming to schema 220.

The outcome of this process is a list of accepted candidates that will be communicated using a message conforming to schema 230. It will be used to construct the list of candidates for each contest.

3.2.2. The Options Nomination Process

This is the process of approving the options to be presented to voters in a referendum. The options can be a straight choice, e.g. YES or NO, to a single question, or can be more complex involving choices to a number of questions and/or preferences of choice.

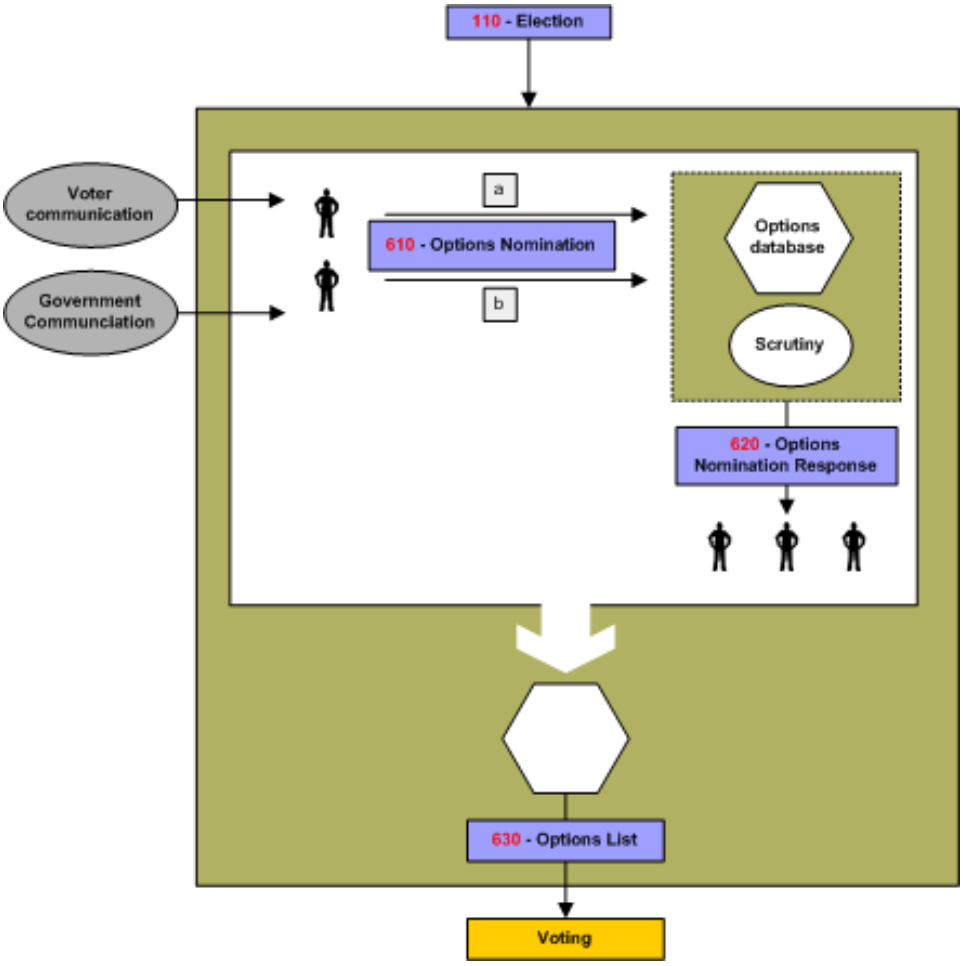


Figure 2D: Referendum Options Nomination Process

The nomination can be received in a number of ways including direct from government institutions or from citizens or businesses, and schema 610 handles the receipt of nominations.

Nominees may be notified of the result of any scrutiny of their nomination using a message conforming to schema 620.

The outcome of this process is a list of accepted options that will be communicated using a message conforming to schema 630. It will be used to construct the list of referendum questions for each contest.

3.2.3. The Voter Registration

This is the process of recording a person's entitlement to vote on a voter registration system. A key part of this process is the identification of the person.

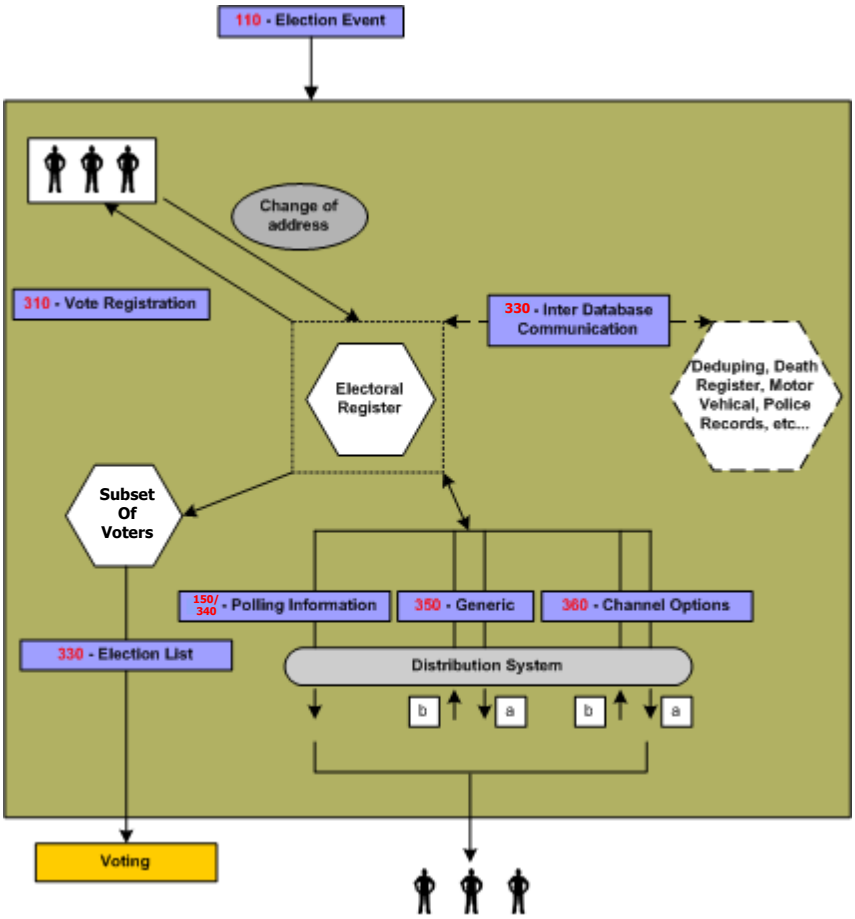


Figure 2E: Voter Registration

The centre of this process is the Electoral Roll Database or the Voters' Database. The input into this database is the outcome of communications between 'a voter' and 'an Election Authority'. The subject of this correspondence can vary from adding a voter to modifying a voter; deletion of a voter is considered as part of modification.

This schema of data exchange is recommended irrelevant of the method a voter uses to supply his information. For example, a voter could register online or simply by completing a voter's form and posting

the signed form. In the latter case, this schema is to be followed when converting the paper form into the electoral database.

Another potential communication or exchange of data is with other databases such as those used by another election authority, government body, etc. Database exchanges will be required in some election scenarios; examples include geographical and organizational boundary changes.

At a certain date, a subset of the voters' database is fixed from which the election list is generated. Schemas contain some subset of the eligible voters, perhaps grouped by polling district or voting channel.

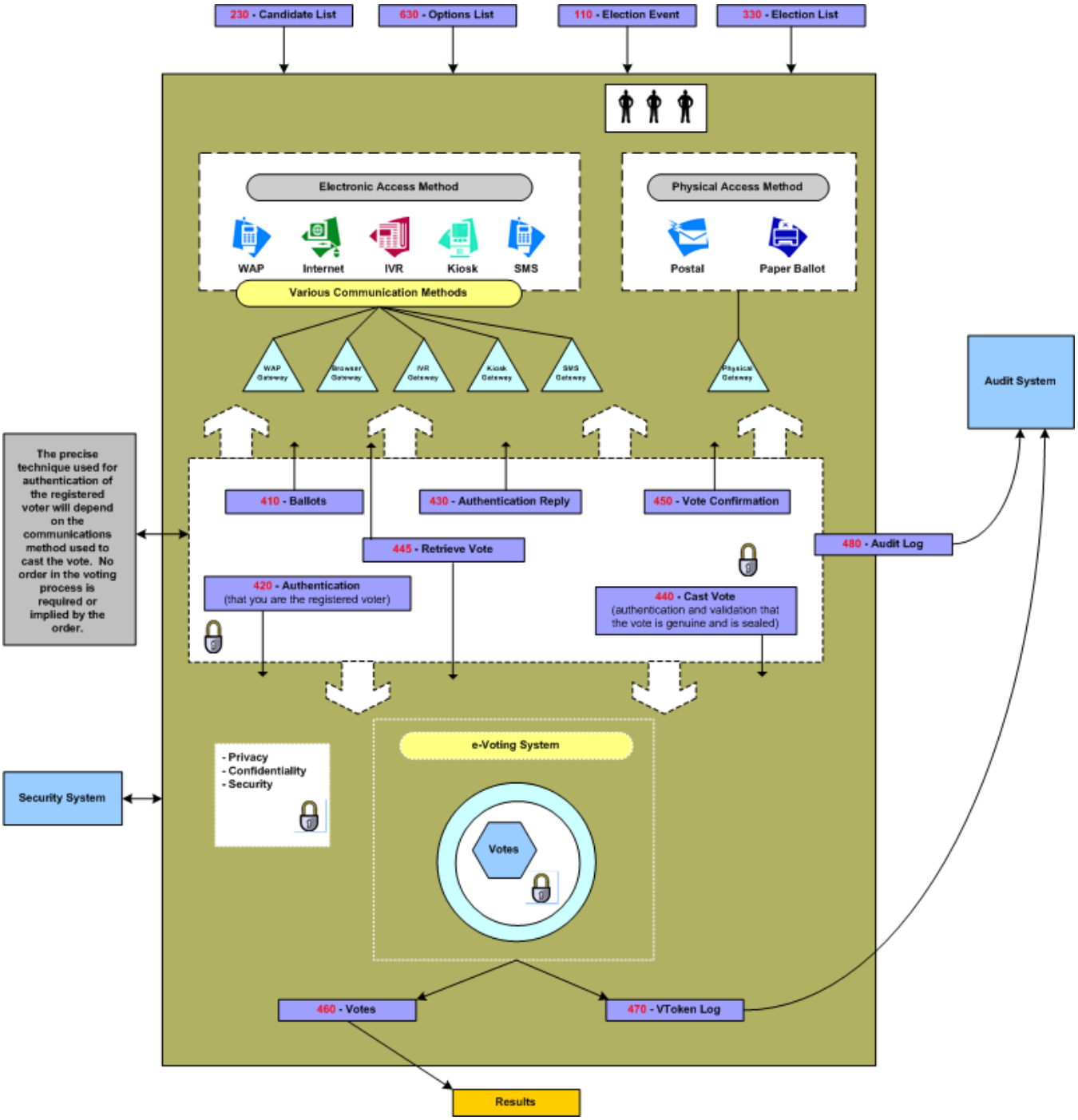
It is here that we introduce the concept of voter communications. Under this category we divided them into three possible types of communications:

- Channel options
- Polling Information
- Generic.

The communication method between the Election Authority and the voters is outside the scope of this document, so is the application itself. This document does specify the data needed to be exchanged.

3.2.4. The Voting Process

This is the process that involves the authentication of the voter and the casting of an individual vote.



381

382 *Figure 2F: The Voting Process*

383

384 We assumed various systems would be involved in providing the voting process and regard each system
385 as an independent entity.

386 As this figure shows, the voter will be voting using a choice of physical channels such as postal or paper
387 ballot (the 'physical access methods'), or the voter can vote using 'electronic access methods' where
388 he/she can utilize a number of possible e-voting channels.

Each channel may have a gateway acting as the translator between the voter terminal and the voting system. Typically, these gateways are in proprietary environments. The following schemas are to be used when interfacing to such gateways: 410, 420, 430, 440 and 450. These schemas should function irrespective of the application or the supplier's favored choice of technology.

When a pre-ballot box is required in a scenario, schema 445 can be used to retrieve and amend votes before they are counted.

Where a voter's right to vote in any particular contest needs to be determined, this is defined by the parameters of his VToken. See Section 4 for more information on security and the VToken.

In some scenarios the right to vote may need to be qualified. This may occur if the voter's right to vote is challenged or if the voter is given the temporary right to vote. In this case the vote needs to be cast by a voter with a Qualified VToken. The reason for the qualification shall always be present in a Qualified VToken and the qualification may need to be investigated before the vote is counted as legitimate. The VToken and Qualified VToken are part of schemas 420, 440, 450, 460 and 470.

To create balloting information, input data is needed about the election, the options/candidates available and the eligible voters; see schemas 230, 110 and 330 for exchanging such information between e-systems.

3.2.5. The Vote Reporting Process

Two of the post election items are the Final or Interim Result and the Audit Report. Audit is discussed in 3.4.6.

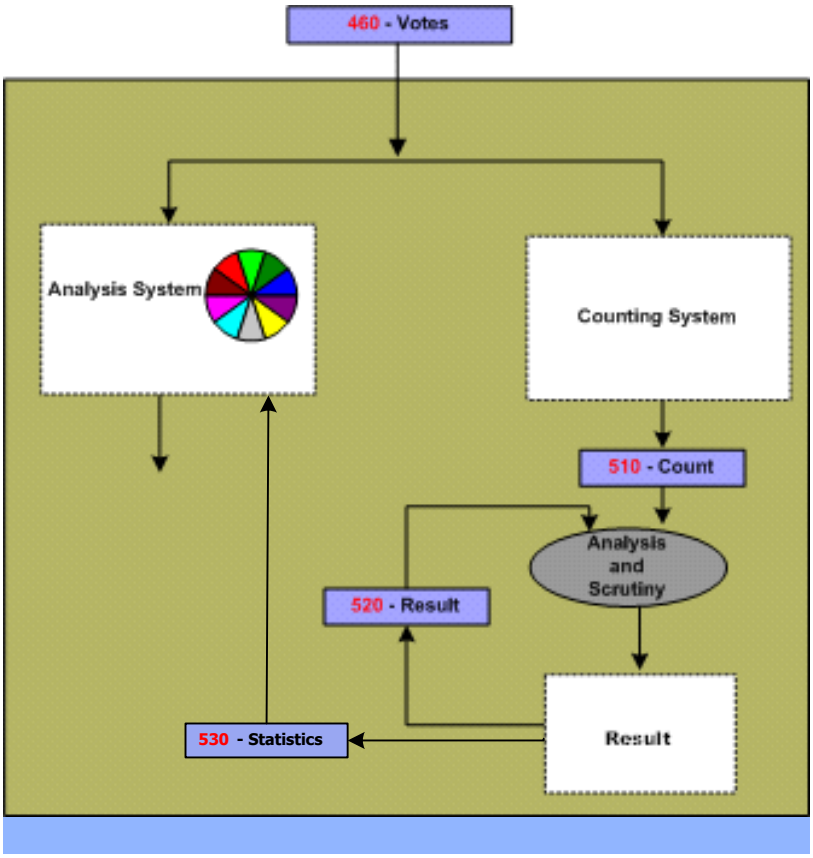


Figure 2G: The Vote Reporting Process

The voting system should communicate a bulk of data representing the votes to the counting system or the analysis system-using schema 460. The count of these, which is the compilation of the 460, is to be communicated by the schema 510.

Recount can be very simply accommodated by a re-run of the schema 460, on the same or another counting system.

Some voting methods, such as the additional member system (AMS), combine the result of one election with the votes of another to create a result. For an election run under the AMS, the results of the 'first past the post' (FPP) election can be communicated using a message conforming to schema 520. This schema can only be used for communicating the results of elections using simple voting methods such as FPP, and is not intended as a general purpose results schema.

The votes schema 460 also feeds into a variety of analysis systems, which can be used to provide for demographic, statistical or other types of election reports. The output of these analysis systems is outside the scope of this document.

Schemas 510 and 520 allow for Simulation and Extrapolation of final or interim Counts and Results. Simulation being the facility to forecast the result of a contest based on the result of another contest. Extrapolation is the facility to forecast the final result of a contest based on the count so far.

Schema 530 allows for a variety of statistics to be extracted from the results and passed to analysis systems and other media outlets.

.

3.2.6. The Auditing System

Audit is the process by which a legal body consisting of election officers and candidates' representatives can examine the processes used to collect and count the vote, thereby proving the authenticity of the result.

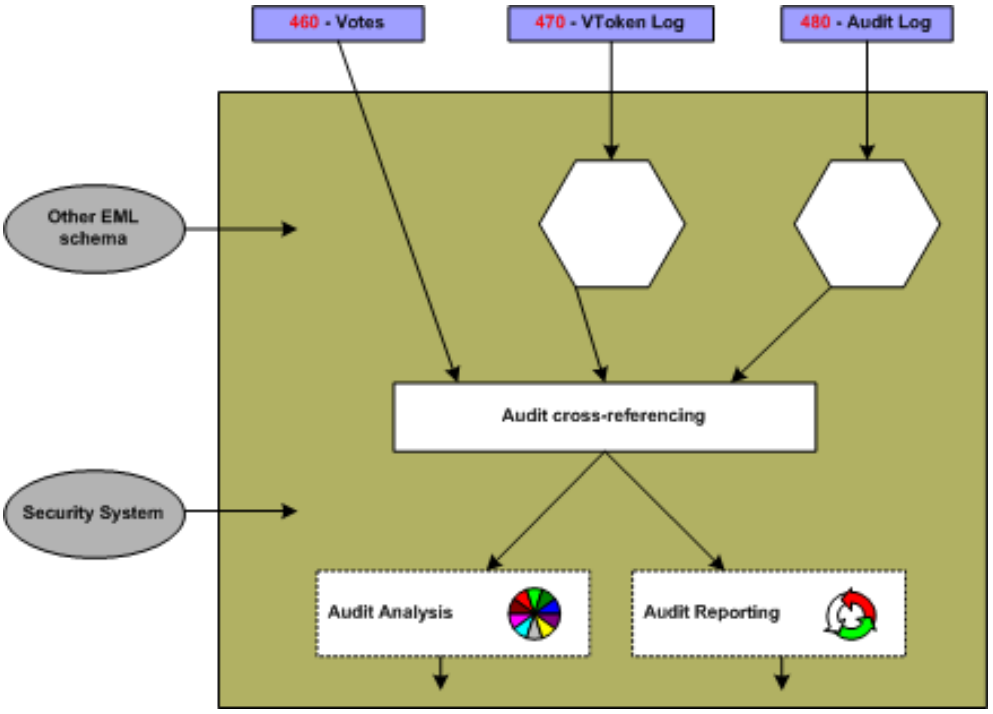


Figure 2H: Auditing System

A requirement is for the election officer to be able to account for all the ballots. A count of ballots issued should match the total ballots cast, spoiled and unused.

Schemas 460, 470, 480 from the voting process provide input data to the audit process. Depending on the audit requirements additional data from other processes may be required. In particular, the security process may provide additional data about all the issued VTokens and Qualified VTokens (see Figure 3A: Voting system security).

The security process ensures that the right to cast a vote is dictated by the presence of a VToken, thus in order to provide accountability for all ballots as per the requirement above, reliable data from the security system is required on the total number of:

- Eligible voters
- Issued VTokens or Qualified VTokens.

The audit process can collate the total number of VTokens and Qualified VTokens provided by the security system with the total number reported by the voting system using schema 460 and 470.

The security system and sealing mechanism should be implemented so that trust can be placed in the seal and hence the sealed data. This implies that the seal should be performed as close to the user submission of the vote as technically possible. The count of the spoiled and unspoiled votes from 460 can then be cross-checked against the count of the number of trusted seals from 480. This correlation confirms that the total number of votes presented by the output of the e-voting system in 460 is consistent with the total number of submitted votes with seals.

The above correlation between trusted data provided by the security process and data provided by the voting process proves that no legitimate votes have been lost by the voting system. It also proves that there is consistency between the number of eligible voters and the spoiled, unspoiled and unused votes as recorded by the e-voting system.

Another requirement is for the election officer to be able to prove that voted ballots received and counted are secure from any alteration. This requirement is met because each vote cast is sealed; the seal can be verified by the audit system and to prove that no alterations have been made since the vote was sealed.

A further requirement is for the election officer to be provided with a mechanism to allow a recount when a result is contested. The number of votes from the voting system using schema 460 can be verified by correlating the total votes as calculated by the audit system (using schema 480), with the totals from the counting system. Then either re-running the count or running the count on another implementation can verify an individual result.

There is also the requirement for the election officer to be provided with a mechanism that allows for multiple observers to witness all the voting process. How this is achieved in dependant on the implementation of the system and procedures adopted. However, the seals and channel information using schema 480 provide the ability to observe voting inputs per channel while voting is in progress without revealing the vote itself or the voter's identity. The final count of the seals can then be used to cross check the totals of the final result as described above.

The above defines some of the election data that can be verified by the audit system. However, ideally everything done by the various components of an election system should be independently verifiable. In the scope of EML this means that the audit system may need to be able to process all the standardized EML schemas. The audit system may in addition support proprietary interfaces of voting systems to enhance visibility and correctness of the election process.

3.3. Data Requirements

Shown below at Fig 2i is a high-level data model of the data used in the above processes. Further lower-level data models are available in the 'EML v6.0 Data Models' file and all the data are defined in 'EML v6.0 Data Dictionary'. Fig 2j below shows the mapping between the data entities and the EML schemas.

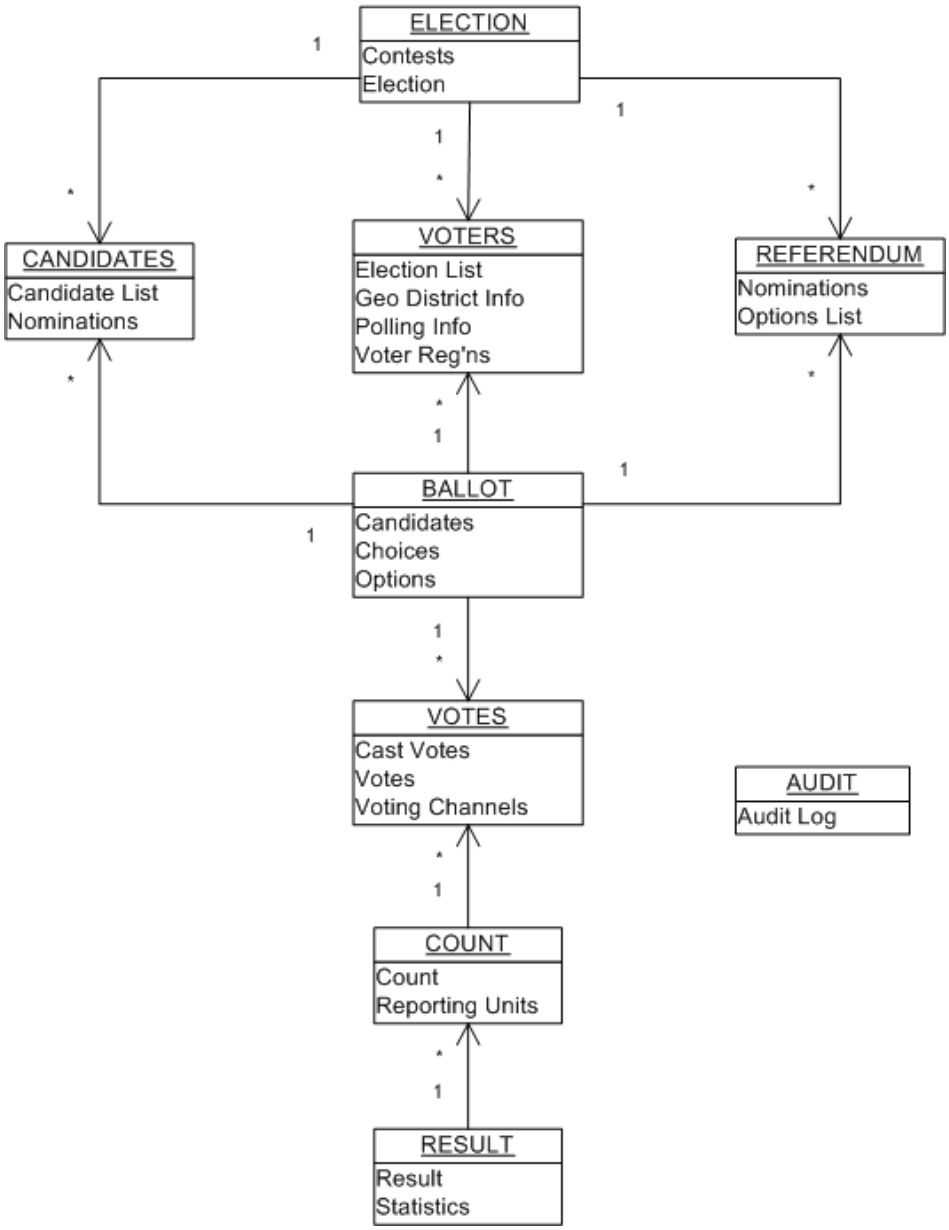


Figure 2i: High-level Data Model

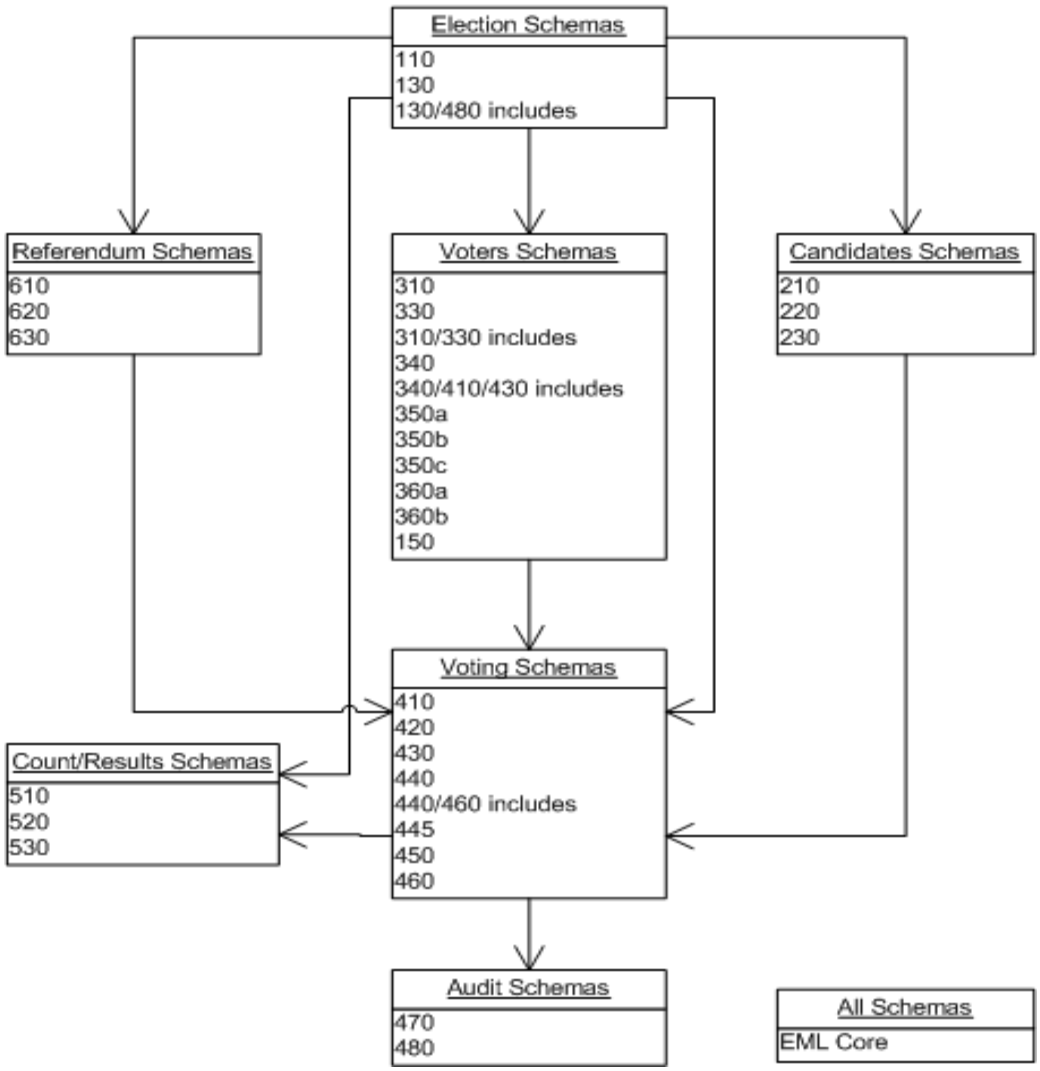


Fig 2j Entity/Schema Mapping

4. Schema Outline

4.1. Structure

The Election Markup Language specification defines a vocabulary (the EML core) and message syntax (the individual message schemas). Thus most voting-related terms are defined as elements in the core with the message schemas referencing these definitions. The core also contains data type definitions so that types can be re-used with different names (for example, there is a common type to allow messages in different channel formats), or used as bases for deriving new definitions.

In some cases, two or more message schemas have large parts in common. For example, a voter authentication response message can contain a ballot that is almost identical to that used in the ballot message. When this occurs, the relevant declarations are included in a file whose file name includes the word 'include' and the number of the schemas in which it is used.

There is a third category of schema document within EML - the EML externals. This document contains definitions that are expected to be changed on a national basis. Currently this comprises the name and address elements, which are based on the OASIS Extensible Name and Address Language [1], but may be replaced by national standards such as those contained in the UK Government Address & Personal Details schemas [2]. Such changes can be made by replacing just this single file.

As well as these, several external schemas are used. The W3C has defined a standard XML signature [5]. OASIS has defined schemas for the extensible Name and Address Language (xNAL) [1]. As part of the definition of EML, the committee has defined a schema for the Timestamp used within EML. All these schemas use their appropriate namespaces, and are accessed using xs:import directives.

Each message (or message group) type is specified within a separate schema document. All messages use the EML element from the election core as their document element. Elements declared in the individual schema documents are used as descendents of the EML element.

As an international specification, EML is generic in nature, and so needs to be tailored for specific scenarios. Some aspects of the language are indicated in EML as required for all scenarios and so can be used unchanged. Some aspects (such as the ability to identify a voter easily from their vote) are required in some scenarios but prohibited in others, so EML defines them as optional. Where they are prohibited, their use must be changed from an optional to prohibited classification, and where they are mandatory, their use must be changed from an optional to required classification.

4.2. Viewing Schemas

EML schemas are supplied as text documents. For viewing the structure of the schemas, we recommend the use of one of the many schema development tools available. Many of these provide graphical displays.

4.3. IDs

XML elements may have an identifier which is represented as an Id attribute.

Each schema element has an Id attribute that relates to the message numbering scheme. Each message also carries this number.

Some items will have identifiers related to the voting process. For example, a voter might be associated with an electoral roll number or a reference on a company share register. These identifiers are coded as elements.

Other identifiers exist purely because of the various channels that can be used for voting (e.g. Internet, phone, postal, etc). In this case the identifiers are likely to be system generated and are coded as attributes.

4.4. Displaying Messages

Many e-voting messages are intended for some form of presentation to a user, be it through a browser, a mobile device, a telephone or another mechanism. These messages need to combine highly structured information (such as a list of the names of candidates in an election) with more loosely structured, often channel-dependent information (such as voting instructions).

Such messages start with one or more Display elements, such as:

```
<?xml version="1.0" encoding="UTF-8"?>
<EML
  Id="410"
  SchemaVersion="6.0"
  xml:lang="en"
  xmlns="http://www.govtalk.gov.uk/temp/voting"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.govtalk.gov.uk/temp/voting
                      ..\schemas\ballot.xs">
  <Display Format="html">
    <Stylesheet Type="text/xsl">../stylesheets/ballot.xsl</Stylesheet>
    <Stylesheet Type="text/css">../stylesheets/eml.css</Stylesheet>
  </Display>
  <Ballots>
    ...
```

This example shows a Display element providing information to the receiving application about an XSL stylesheet which transforms the message into HTML for displaying the ballot in a Web browser. In the Display element in the example, the XSLT stylesheet reference is followed by a CSS stylesheet reference. In this case, the XSLT stylesheet referenced will pick up the reference to the CSS stylesheet as it transforms the message, and generate appropriate output to enable the displaying browser to apply that cascading stylesheet to the resulting HTML.

Not all information in a message will need to be displayed, and the creator of the message might have views on the order of display of the information. To allow stylesheets to remain generic, many elements in the schemas can have a DisplayOrder attribute. The values of these attributes determine the layout of the display (or the spoken voice if transforming to, for example, VoiceXML), even when using a generic stylesheet.

When displaying messages in HTML, the expectation is that generic stylesheets will cover most cases, with the stylesheet output being embedded in a web page generated from an application-specific template. Similarly, voice applications might have specific welcome and sign-off messages, while using a generic stylesheet to provide the bulk of the variable data.

The three screen shots show the effect of using the same XSL stylesheet on the ballots for various voting scenarios. In the first picture, clicking on the name of a candidate has popped up a window with additional details.

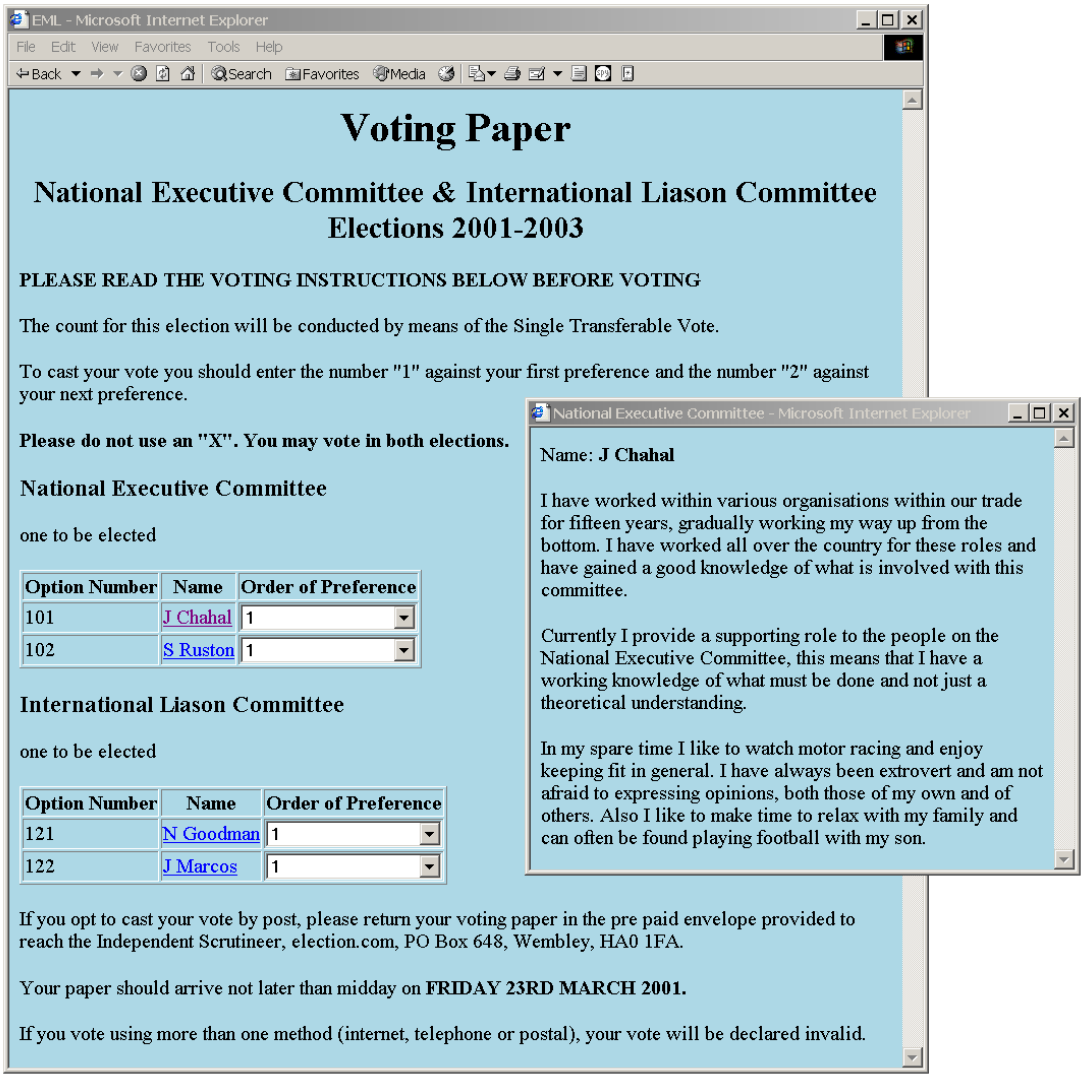
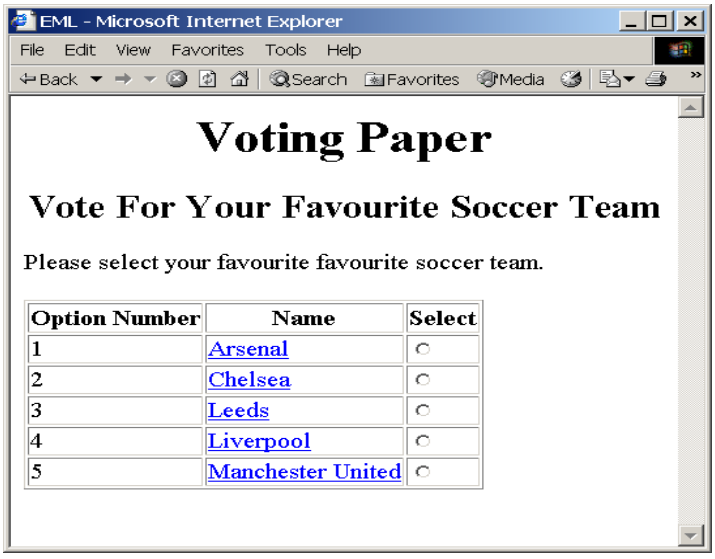


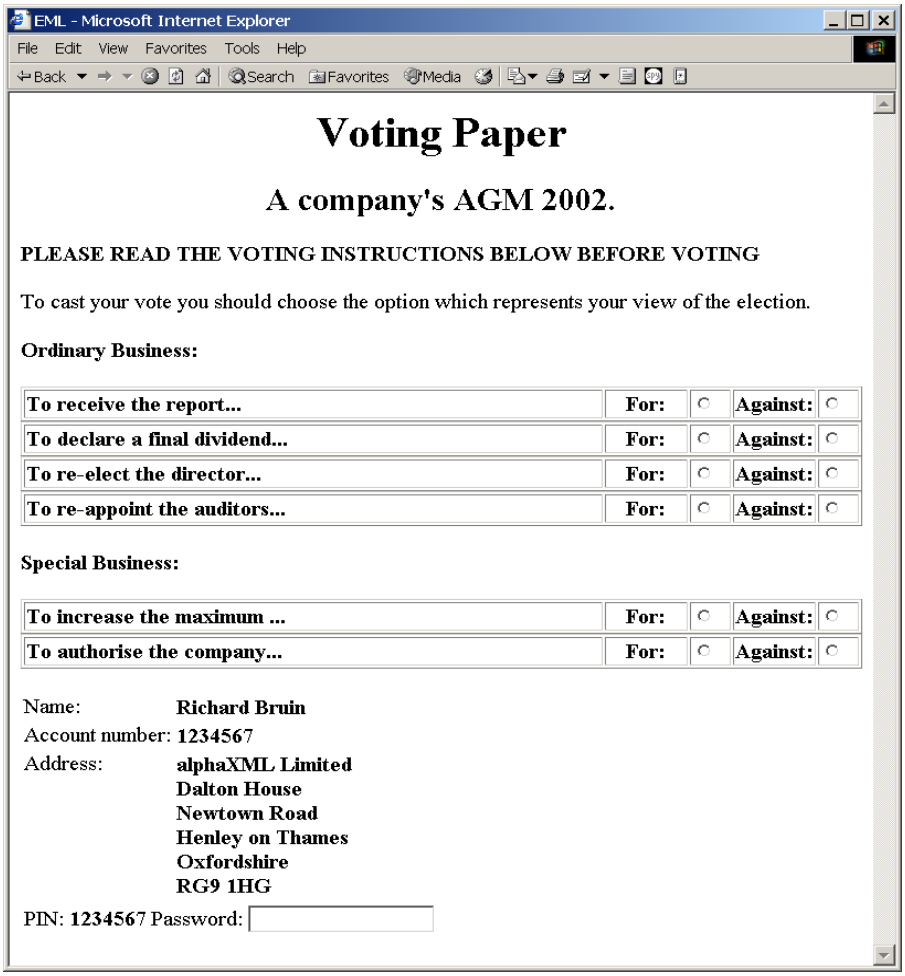
Figure 3A: Screen shot of the ballot for scenario 1



576

577 *Figure 3B: Screen shot of the ballot for scenario 2*

578



579

580 *Figure 3C: Screen shot of the ballot for scenario 3*

4.5. EML Message Validation

It is up to each specific system implementation whether it uses these schemas for validation of EML messages for either testing or live use. The recommended approach is to validate incoming messages against the EML schemas (with the application-specific EML externals schema), then further validate against the relevant Schematron schema or OASIS CAM template. The first stage requires the use of an XML processor (parser) that conforms to W3C XML Schema. The second stage requires either an XSLT processor or a dedicated Schematron or CAM processor.

However, an implementation may choose to:

- modify the EML schemas to incorporate those application-specific constraints that can be represented in W3C XML Schema;
- not validate the rules that are encoded as templates schemas (Schematron or CAM);
- not perform any validation; or
- develop some alternative backend validation.

4.6. Namespaces

The message schemas and the core schema are associated with the namespace `urn:oasis:names:tc:evs:schema:eml`. This is defined using the prefix `eml`. The XML Schema namespace `http://www.w3c.org/2001/XMLSchema` is identified by the prefix `xs` and the XML Schema Instance namespace `http://www.w3.org/2001/XMLSchema-instance` by the prefix `xsi`.

Use is also made of namespaces for the Extensible Name and Address Language (xNAL). The Extensible Name Language namespace `urn:oasis:tc:ciq:xsdschema:xNL:3.0` is identified by the prefix `xNL`, and the Extensible Language namespace `urn:oasis:names:tc:ciq:xsdschema:xAL:3.0` by the prefix `xAL`.

4.7. Extensibility

Various elements allow extensibility through the use of the `xs:any` element. This is used both for display information (for example, allowing the sending of HTML in a message) and for local extensibility. Note that careless use of this extensibility mechanism could reduce interoperability.

4.8. Additional Constraints

The EML schemas provide a set of constraints common to most types of elections worldwide. Each specific election type will require additional constraints, for example, to enforce the use of a seal or to ensure that a cast vote is anonymous. It is recommended that these additional constraints be expressed using the Schematron language although other validators, e.g. OASIS CAM, can be used. This allows additional constraints to be described without altering or interacting with the EML schemas. Any document that is valid to a localization expressed in Schematron must also be a valid EML document.

4.9. Metadata

Some messages need information relating to the issuing of them, such as the issue date, who issued them etc. This is most likely to be a requirement for the 330 message but is equally applicable to 130, 230, 350a and several others. For that reason, it is useful to make this optional information available in the header. The information usually consists of: managing authority, date of issue, start of list period (used for changes to the list to indicate the start of the period for which changes are being shown), end of list period (i.e. the date of the snapshot of the list).

4.10. Splitting of Messages

There is sometimes a need to split long messages into several parts. By their nature, each of these messages will contain a small amount of background information and a single element type that is repeated many times. For example, the 330-electionlist message can have many VoterDetails elements.

When a message is split, each part must be a complete, valid EML document. This will contain all the elements required by EML and the specific application. Those parts outside the repeated element that relate to the message as a whole, such as the TransactionId, must have the same values in each part message. The values of those elements and attributes that relate to an individual part message, such as the SequenceNumber, may vary between the individual part messages. Information in the EML element indicates the sequence number of the message and the number of messages in the sequence. Each message in the sequence must contain the same TransactionId, and must indicate the repeated element according to the table below. Only the messages shown in the table may be split in this way.

Message	Repeated Element
330-electionlist	VoterDetails
340-pollinginformation	Polling
410-ballots	Ballot
460-votes	CastVote
470-vtokenlog	VTokens
480-auditlog	LoggedSeal

For ease of implementation, a message that can be split may contain the elements used for splitting even if the entire message is sent in one piece. In this case, the values of SequenceNumber and NumberInSequence will both be "1".

4.11. Error Messages

The 130 schema is used to define a message for reporting errors in EML messages.

Error messages are given codes. These fall into one of five series:

1000	XML well-formedness or Schema validation error
2000	Seal error
3000	EML rule error
4000	Localization rule error
5000	System specific error

If the error type is not message-specific (or is a general rule applying to several schemas), the series reference above is used. If it is message-specific, the last three digits of the error series (and possibly a final alpha character) reflect the message type. A three digit error code is appended to the series code, separated by a hyphen.

An error code relating to a localisation applicable to all message types could therefore be 4000-001. One specific to the localization of schema 110 could be 4110-002.

4.12. All Schemas

4.12.1. XML well-formedness or Schema validation error

Error code	Error Description

1000-001	Message is not well-formed
1000-002	Message is not valid

649 ### 4.12.2. Seal Errors

Error code	Error Description
2000-001	The Seal does not match the data

650 ### 4.12.3. EML Additional Rules

651 The following rules apply to messages regardless of localization. One of the two rules on splitting will
652 apply to each message type as described in the table below.

Error Code	Error Description
3000-001	If there are processing units in the `AuditInformation`, one must have the role of sender
3000-002	If there are processing units in the `AuditInformation`, one must have the role of receiver
3000-003	This message must not contain the elements used for splitting
3000-004	The value of the `Id` attribute of the EML element is incorrect
3000-005	The message type must match the `Id` attribute of the `EML` element
3000-006	All messages that are split must include the correct sequenced element name.

653

	3000-003	3000-006
110	✓	
130	✓	
150	✓	
210	✓	
220	✓	
230	✓	
310	✓	
330		✓
340		✓
350a	✓	
350b	✓	
350c	✓	

360a	✓	
360b	✓	
410		✓
420	✓	
430	✓	
440	✓	
445	✓	
450		✓
460		✓
470		✓
480		✓
510	✓	
520	✓	
530	✓	
610	✓	
620	✓	
630	✓	

654

5. Schema Descriptions

5.1 Overview

The following table presents a high-level overview of the EML schemas. Further explanations are given in the following sub-paragraphs.

Schema Name	Purpose
EML 110 – election event	Information about an election or set of elections. It is usually used to communicate information from the election organizers
EML 130 – response	Report error response. Contains details of the message received that was in error.
EML 150 – geographic district	Allow use of geographic mapping systems to describe the election districts and boundaries and balloting
EML 210 – candidate nomination	Used to nominate candidates or parties, consenting or withdrawing
EML 220 – response to nomination	Use to confirm whether the candidate's nomination has been accepted.
EML 230 – candidate list	Contest and candidates details
EML 310 – voter registration	Used to register voters for an election
EML 330 – voter election list	Details of actual voters for an election
EML 340 – polling information	Notification to voter of an election, their eligibility and how to vote
EML 350a – outgoing generic	Provides a common structure for communications to the voter.
EML 350b – incoming generic	Provides a common structure for communications from the voter.
EML 350c – internal generic	Provides a common structure for systems communications.
EML 360a – outgoing channel	Used for messages offering a set of voting channels to the voter
EML 360b – incoming channel	Used for messages defining a preferred voting channels of the voter
EML 410 – ballot	Describes the actual ballot to be used for an election
EML 420 – voter authentication	Used for voter authentication during a voting process
EML 430 – authentication response	Indicates whether authentication succeeded; may present the ballot to the user
EML 440 – cast vote	Actual record of vote cast
EML 445 – retrieve vote	For systems that include a pre-ballot box from which votes can be retrieved and confirmed
EML 450 – confirm vote	Show whether a vote has been accepted and provide a reference number or rejected.
EML 460 – votes group	Group of votes being transferred for counting
EML 470 – vtoken log	Add voting tokens to an audit log
EML 480 – audit log	Documents access to voting records and reason
EML 510 – count	Results of election contest(s) and counts
EML 520 – result	Communicating specific result details on candidates and elections
EML 530 – statistics	Provide statistical information about EML 510 counts and results
EML 610 – options nomination	Used to nominate the choice of options that will be included in a referendum.

EML 620 – options nomination response	Confirms whether the options nomination has been accepted.
EML 630 – options list	Use to transfer lists of proposals for a referendum
EML Core	Defines the core definitions of the content model reused across the EML schemas

659

5.2 EML Core Components

The EML Core schema contains elements and data types that are used throughout all the EML schemas.

For details see the EML core dictionary that is provided as separate files in XML and spreadsheet formats. The core components are included in the EML Core schema that is imported into each EML schema.

The dictionary shows items in sequence and denotes their CCTS (Core Components Technical Specification) classification based on their usage within EML structures. Those marked as BBIE (Basic Business Information Entity) are atomic pieces (element), while those marked as ABIE are Aggregate entities consisting of more than one component (elements structure), while ASBIE equate to XML attributes values for the associated BBIE elements. For complete discussion of Core Components concepts see the UN/CEFACT Core Components specification (http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf).

Related to classification of content type is the difference between Schema elements and types and specifically Schema xsd:complexType usage and this is discussed next.

5.2.1 Complex Data Types

The choice between defining an element or a data type for a reusable message component is a significant design issue. It is widely accepted as good practice to use element declarations when there is good reason to always refer to an element by the same name and there is no expectation of a need to derive new definitions. In all other cases, data type declarations are preferable. The term schema component is used to refer to elements and data types collectively.

When defining a complete mark-up language, limiting the use of elements and types can restrict further development of the language. For that reason, both data types and elements are defined in EML. Only where an element is an example of a primitive or derived data type defined in XML Schema Descriptions is no explicit data type defined within EML.

In use, it is expected that, for example:

- A voting token will always have an element name VToken and so will use the element name.
- A logo or a map have similar definitions, so both use the PictureDataStructure. There is no PictureData element.
- Within voter identification, some elements will usually need to be made mandatory and so a schema will specify a new element based on the VoterIdentificationStructure data type.

5.3 Message Schemas

This section describes the EML messages and how the message specifications change for this application. It uses the element and attribute names from the schemas.

Election Event (110)

This schema is used for messages providing information about an election or set of elections. It is usually used to communicate information from the election organisers to those providing the election service.

The message therefore provides information about the election event, all elections within that event and all contests for each election.

For the election event, the information includes the ID and name of the event, possibly with a qualifier on the event. This qualifier is used when an event has several local organisers. For example, for a UK general election, each constituency organises its own contests. The election event is therefore the general election, whilst the qualifier would indicate the constituency. Other information regarding an election event comprises the languages to be used, the start and end dates of the event, potentially a list of external documents that are applicable (such as the rules governing the election), a description and information about the managing authority.

The managing authority can be indicated for the event, each election, each contest within the election and each reporting unit.

An election can have a number of dates associated with it. For example, there is likely to be a period allowed for nomination of candidates and a date when the list of eligible voters is fixed. Each date can be expressed as a single date when something happens, a start date, an end date, or both start and end dates. These dates can be either just a date or both a date and time using the subset of the ISO 8601 format supported by XML Schema.

Like the event, an election can have both a managing authority and referenced documents. Finally, there is a `Messages` element for additional information.

A contest has a name and ID. It can also have reporting unit identifiers. A contest may need to specify its geographical area independently from its name, for which purpose the Area element is provided. Each contest can specify the voting channels allowed. In general, the list of possible channels will be further restricted as part of a local customisation. Each channel can specify several methods for authenticating the voter, such as PIN and password, and a response method, indicating the type of response to be given to a cast vote. Finally, facilities are provided to indicate the dates and times when the channel will be available to the voter.

As described previously, a contest can indicate its managing authority. It may also indicate the position (such as 'President') for which votes are being cast. The Description allows for additional text describing the contest. Each contest indicates the voting method being used, whilst the `CountingAlgorithm` indicates the method of counting (such as the d'Hondt or Meeks method) that will be used. The minimum and maximum number of votes to be cast by each voter can also be indicated.

A list of polling places can be provided. These can be either physical locations for people to go to vote, postal addresses for postal votes or electronic locations. An 'other location' is also allowed for cases where these do not meet the requirements. A location can also say when it will be available. This is intended for mobile polling stations that will only be available at a given address for a part of the voting period.

Finally, a `Messages` element allows for additional information that might be communicated to the voter later through other messages.

Additional Rules

Error Code	Error Description
3110-001	The allowed channels must not be declared at both the election event level and the contest level.

Response (130)

Some messages have a defined response message that provides useful information. However, there is a need for a more general response, either to indicate that a message has been accepted, or to indicate the reasons for rejection.

The message includes information to identify the message to which the response applies (by using the same transaction id in the `EML` element and, if necessary, including the sequence number of the message to which the response applies in the Response element), with information on the entity raising the message, whether the message was accepted and information about the errors if it was not. The desired language for a display message can also be included to allow a downstream processor to substitute a language-specific error message if required.

If the message is reporting an error, the location of the error within the message can be indicated. Usually, this will be an XPath to the location of the error. However, errors detected by an XML parser may be in a different format, such as a line number.

Note that a single response can be raised for a series of sub-messages with the same transaction ID. This allows indication, for example, that a sub-message was missing.

Additional Rules

Error Code	Error Description
3130-001	If the message is not accepted, there must be an Errors element

GeoDistrict (150)

This schema allows the use of geographic mapping systems to describe election districts and their boundaries by providing information to voters to help their understanding of where and when they should go to cast their vote. For example information relating to the streets and polling places within a district, the name by which the district is identified to voters and physical features and landmarks describing a specific polling place to be used in elections.

Supplementary information about the districts and polling places to assist voters can also be recorded, for example detailed descriptions in one or more language that describes the district, the political area or legislature that the district belongs to, the Authority that is responsible for managing elections in the district, and access and facilities details about a specific polling place to be used in elections.

This set of authorized information can be made available by any number of organisations through a variety of different outlets.

Candidate Nomination (210)

Messages conforming to this schema are used for four purposes:

1. nominating candidates in an election;
2. nominating parties in an election;
3. consenting to be nominated; or
4. withdrawing a nomination.

Candidate consent can be combined in a single message with a nomination of the candidate or party or sent separately.

Note that the message does not cover nomination for referendums.

The election and contest must be specified. When a candidate is being nominated, there must be information about the candidate and one or more proposers. The candidate must supply a name. Optionally, the candidate can provide contact information, an affiliation (e.g. a political party) and textual profiles and election statements. These two items use the `MessagesStructure` to allow text in multiple languages. There is also scope to add additional information defined by the election organiser.

The proposers use the standard proposer declaration with a mandatory name and optional contact information and job title. Again, additional information can be required.

If a party is being nominated, the primary proposer will be the contact. Information on candidates in a party list can also be provided.

Candidates, either individuals or on a party list, must define the action being taken and may provide scrutiny information. The scrutiny requirements indicate how the candidate has met any conditions for standing in this election. This could include indicating that a deposit has been paid or providing a reference to prove that he or she lives in the appropriate area. This information can be signed independently of the complete message.

Response to Nomination (220)

This message is sent from the election organiser to the candidate or nomination authority for a party to say whether the nomination has been accepted. Along with the acceptance information and the basic information of election, contest and party and candidate names, the candidate's contact details and affiliation can be included and a remark explaining the decision.

Additional Rules

Error Code	Error Description
3220-001	If the nomination has not been accepted, a reason for rejection is required in the Remark element

Candidate List (230)

This schema is used for messages transferring candidate lists for specified contests. It has the election event, election and contest identifiers, and optionally the event dates and a contest description. The list itself can be either a list of candidates, each with a name, address, optional affiliation and other useful data, or a list of parties. In the latter case, contact information and a list of candidates under a party list system can also be included.

Voter Registration (310)

This schema is used for messages registering voters. It uses the `VoterIdentificationStructure`. The `VoterInformationStructure` is used unchanged. Proof of ID can be provided.

There is the facility for the transmission channel (for example a trusted web site) to add the time of transmission.

Additional Rules

Error Code	Error Description
3310-001	The Proxy must not have a VToken or VTokenQualified

Election List (330)

This schema is primarily used for messages communicating the list of eligible voters for an election or set of elections. It can also be used for any other purpose that involves the transfer of voter information. Partial lists are allowed through the use of the Qualifier, Blocked and VoterGroup elements. So, for example, a list of postal voters or a list of proxies can be produced. The schema can also be used for filtered lists such as a list of postal proxies. These lists sometimes do not contain any names meeting the filter so empty lists are allowed.

For each voter, information is provided about the voter himself or herself, and optionally about the elections and contests in which the voter can participate. The information about the voter is the same as that defined in the 310-voterregistration schema. Added to this can be a list of elections, each identifying the election and the contest in which this voter is eligible to vote, and the polling places available. Any voter can have a Blocked element set against them with an optional Reason and Channel. This allows a list to be produced for a polling place indicating those that have already voted by another means or who have registered for a postal vote. It can also be used if the complete electoral register must be transmitted (perhaps as a fraud prevention measure) but some people on the register are no longer eligible to vote.

Additional Rules

Error Code	Error Description
3330-002	The polling district can only be included for either the voter or the election.
3330-003	The polling place can only be included for either the voter or the election.

Polling Information (340)

The polling information message defined by this schema is sent to a voter to provide details of how to vote. It can also be sent to a distributor, so multiple sets of information are allowed. In the case of SMS voting, ballot information may also be required, so this can be included. Either one or several sets of polling information may be sent to each voter for any election event.

Some information about the voter and any proxy may be included, for example to print on a polling card. This can also include a mailing address for a distributor to use.

Information about the elections and contests is included for the benefit of the voter. For each voting channel, this includes where to vote (which could be a polling station, address for postal voting, URL for Internet voting, phone number for SMS voting etc) and the times that votes can be placed. Use of the DisplayOrder attribute on these allows the display or printing of information to be tailored from within the XML message.

Ballot information may be included if required. This is a subset of the information defined in the 410-ballots schema. In this case, it is likely that the short code for a candidate will be used for SMS voting. It is possible that an expected response code will be provided as well. Both the short code and expected response code may be tailored to the individual voter as part of a security mechanism.

Outgoing Generic Communication (350a)

This schema provides a common structure for communications to the voter. Individual message types can be designed based on extensions of this schema.

The voter must always provide a name and might provide one or more identifiers. These are shown as a restriction of the VoterIdentificationStructure, the restriction being to leave out the VToken and VTokenQualified. Contact details are also required, and it is expected that at least one of the allowed contact methods will be included. Inclusion of proxy information is optional.

The identifiers for the election event, election and contest are optional. There is then an element in which a message can be placed in any of several different formats according to the channel being used.

Incoming Generic Communication (350b)

This schema provides a common structure for communications from the voter. Individual message types can be designed based on extensions of this schema.

The voter's name must be provided and there can be one or more identifiers. These are shown as a restriction of the `VoterIdentificationStructure`, the restriction being to leave out the `VToken` and `VTokenQualified`. Contact details are also required, and it is expected that at least one of the allowed contact methods will be included. Inclusion of proxy information is optional.

The identifiers for the election event, election and contest are optional. There is then an element in which a message can be placed in any of several different formats according to the channel being used.

Internal Generic (350c)

This schema provides a common structure for communications between those involved in organizing an election. Individual message types can be designed based on extensions of this schema.

There are optional `To` and `From` elements, which can contain any EML elements. It is expected that these will usually be a responsible officer or a person's name and contact information.

The identifiers for the election event, election and contest are optional. There is then an element in which a message can be placed in any of several different formats according to the channel being used.

Outgoing Channel Options (360a)

This schema is used for messages offering a set of voting channels to the voter. It is an extension of schema 350a. A message conforming to this schema will include a list of allowed channels, either to request general preferences or for a specific election event or election within the event.

Incoming Channel Options (360b)

This schema is used for messages indicating one or more preferred voting channels. It may be sent in response to 360a or as an unsolicited message if this is supported within the relevant jurisdiction.

It is an extension of schema 350b, and indicates preferred voting channels in order of preference.

Ballots (410)

This schema is used for messages presenting the ballot to the voter or providing a distributor with the information required to print or display multiple ballots.

In the simplest case, a distributor can be sent information about the election event and a ballot ID to indicate the ballot to print.

In other cases, the full information about the elections will be sent with either an election rule ID to identify the voters to whom that election applies or a set of voter names and contact information. If the ballot is being sent directly to the voter, this information is not required. Since printed ballot papers are likely to require a unique identifier printed on them, the range to be used for each ballot type can be defined.

The election information starts with the election identifier and description. This is followed by information related to the contest and any other messages and information required. Note that each voter can only vote in a single contest per election, so only a single iteration of the Contest element is required.

A contest must have its identifier and a list of choices for which the voter can vote. A voter can vote for a candidate, an affiliation (possibly with a list of candidates) or a referendum proposal. There is also a set of optional information that will be required in some circumstances. Some of this is for display to the voter (`HowToVote` and `Messages`) and some controls the ballot and voting process (`Rotation`, `VotingMethod`, `MaxVotes`, `MinVotes`, `MaxWriteIn`).

Authentication (420)

The authentication message defined by this schema may be used to authenticate a user during the voting process. Depending on the type of election, a voter's authentication may be required. The precise mechanism used may be channel and implementation specific, and can be indicated using the

LoginMethod element. In some public elections the voter must be anonymous; in which case the prime method used for authentication is the voting token. The voting token can contain the information required to authenticate the voter's right to vote in a specific election or contest, without revealing the identity of the person voting. Either the VToken or the VTokenQualified must always be present in an authenticated message. The VotingChannel identifies the channel by which the voter has been authenticated.

Authentication Response (430)

The authentication response is a response to message 420. It indicates whether authentication succeeded using the Authenticated element, and might also present the ballot to the user. This is a restriction of the Ballots element to allow only a single ballot per reply.

Cast Vote (440)

This message represents a cast vote, which comprises an optional voting token (which may be qualified) to ensure that the vote is being cast by an authorized voter, information about the election event, each election within the event and the vote or votes being cast in each election, an optional reference to the ballot used, the identifier of the reporting unit if applicable and a set of optional audit information.

For each election, the contest is identified, with a set of, possibly sealed, votes. The votes are sealed at this level if there is a chance that the message will be divided, for example so that votes in different elections can be counted in different locations.

The selection of candidates, affiliations or a referendum option uses the Selection element. If an election requires preferences to be expressed between candidates, multiple Selection elements will be used, each of these having a suitable Value attribute. Some elections allow write-in candidates, and these are handled in a similar way. Preferences can also be expressed between parties, using the Affiliation element. The PersonalIdentifier is used in elections where each voter is given an individual list of codes to indicate their selection.

A more complex election might request the voter to vote for a party, then express a preferences of candidates within the party. In this case, the Affiliation element is used to indicate the party selected, and multiple CandidateIdentifier elements, each with a Value attribute are used to express candidate preferences.

Preferences in a referendum are handled in the same way as they are for candidates and parties, using the ReferendumOptionIdentifier.

Retrieve Vote (445)

This message is used for voting systems that include a pre-ballot box from which votes can be retrieved and amended before being counted. When a vote is retrieved, it should be deleted from the pre-ballot box.

Vote Confirmation (450)

The vote confirmation message can be used to show whether a vote has been accepted and provide a reference number in case of future queries. Some voting mechanisms require multiple ConfirmationReference elements. If the vote is rejected, the Remark element can be used to show a reason.

Votes (460)

This schema is used to define a message comprising a set of votes being transferred for counting. It is a set of CastVote elements from schema 440 with the addition of the ProposedRejection and ProposedUncounted elements and audit information for the voting system. If a vote is rejected, for example, because a voter has chosen to spoil a ballot paper, many authorities will want to count that vote

as having been cast. The UncountedVotes element is reserved for those cases where that record is not required, for example when the result is thought to be fraudulent. A ProposedRejection or ProposedUncounted element must have a ReasonCode attribute, and may have a Reason attribute to describe the code. They may also have an Objection attribute. This indicates that someone has objected to this vote being rejected or the proposal that it should not be counted.

VToken Log (470)

The message defined by this schema is used to add voting tokens (which may be qualified) to an audit log. The VToken or VTokenQualified is extended by the addition of a Status attribute with a value of voted or unvoted for the VToken and voted, unvoted and withdrawn for the VTokenQualified. In addition to sending single tokens as they are used, the schema can be used to validate a message sending multiple tokens optionally grouped by voting channel. This might be used instead of sending tokens as they used or, for example, to send the unused tokens at the end of an election. The Update element can be used to indicate that an existing log is being updated rather than the message containing a complete new log. The logging system can also be identified for audit purposes.

Audit Log (480)

The message defined by this schema is used to log the use of each seal with associated information for audit purposes.

An audit log message can be transmitted individually as the message causing the log entry is sent or received, or the logs can be stored, and several seals logged at once. Ideally, every device that can create or consume a message will create a log entry so that pairs of entries can be matched. The most important messages to log are those associated with the voting process itself, and these are shown below.

When used in this message, the Response element will not have an AuditInformation child.

	Originating Device	*Gateway*	*Voting System*	*Counting System*	*Vtoken Logging System*	*Seal Logging System*	*Other*	*Notes*
130								4
410	next receiver	receiver	sender					
420	previous sender	sender	receiver					
430	next receiver	receiver	sender				sender / receiver	3
440	previous sender	sender	receiver					
445	previous sender	sender	receiver					
450	next receiver	receiver	sender					
460			sender	receiver				
470			sender	sender	receiver		sender	
480	sender	sender	sender	sender	sender	receiver	sender	2
510				sender			receiver	
520				sender			sender / receiver	

Notes:

1. In some cases (e.g. a kiosk) there may be no gateway involved. In this case, the values in the Gateway column apply to the Originating Device.
2. Creators and receivers of 480 (audit log) messages may not be required to log the seals. In particlar, if an adit log message is sent per seal created or received, the seal on the 480 message must not be logged.
3. "Other" may be the sender when the message is sent to a printer. In this case, the receiver will also be an "Other".
4. An audit log should only be created when the message is used to communicate an error. Most devices can send or receive 130 messages.

The message may contain the name and ID of the event, election and contest. It can also indicate whether this is an update to an existing log or a new log. Following the logged seals, a text message can be added as well as audit information for the audit logging message itself.

Each seal being logged must indicate whether the device sending the log was the sender or receiver of the sealed message. It may be accompanied by the voting token associated with the seal and possibly additional audit information. This will be the audit information from the message being logged with additional information about the message. Most of this is common to all message types, but some message types require specific audit information. One of these is the 130-response message. When this is used to convey an error, almost the complete message payload (the `Response` element and its contents apart from the audit information) is logged with the usual message-independent data.

Count (510)

The count message defined by this schema is used to communicate the results of one or more contests that make up one or more elections within an election event. It may also be used to communicate the count of a single reporting unit for amalgamation into a complete count.

The message includes the election event identifier, and for each election, the election identifier, an optional reference to the election rule being used and information concerning the set of contests.

In some cases, reporting for a contest may be required at a lower level (for example, for each county in a state). For this reason, reporting may be done at the level of the reporting unit, the total votes, or for a total vote and the breakdown according to the multiple reporting units.

Each contest indicates its identifier, and optionally the counting system and the maximum number of votes that each voter could cast. The key information is that about the votes cast for each of the choices available and the numbers of abstentions and rejected and uncounted votes. If a vote is rejected, for example, because a voter has chosen to spoil a ballot paper, many authorities will want to count that vote as having been cast. The `UncountedVotes` element is reserved for those cases where that record is not required, for example when the result is thought to be fraudulent. Both the `UncountedVotes` and `RejectedVotes` elements have `Reason` (optional) and `ReasonCode` (mandatory) attributes to indicate why the votes were treated as they have been. The former is a textual description, and the latter a code.

For each choice available to the voter, the identifier and number of valid votes are mandatory. The other information provided depends on the type of election. For example, the `Value` attribute of the `Selection` element can be used to indicate whether a candidate was a first or second choice in an election run under the single transferable vote system. In the simplest cases, the identifier for the candidate (perhaps with the party), the party or the referendum option is given. If the voter was able to vote for a party and provide a preference for candidates within the party, the `AffiliationIdentifier` element is used, and multiple `CandidateIdentifier` elements may be used, each with a `Count` attribute. This count is the result of whatever algorithm has been used to calculate the ranking of the candidates.

This schema allows for Simulation and Extrapolation of Counts and subsequently Results. Simulation being the facility to forecast the result of a contest based on the result of another contest. Extrapolation is the facility to forecast the final result of a contest based on the count so far.

Result (520)

Messages described by this schema can be used to communicate the results of simple election types. One specific use is to provide an input into the calculation algorithm for elections using the additional member system.

The main part of the schema is held within the `Selection` element. This allows a choice of candidate, affiliation or referendum option identifiers to be defined with the position that choice achieved (first, second etc). Optionally, the number of votes can be shown. A candidate can be associated with his or her affiliation if required. Write in candidates will be shown in the same way as other candidates, although they will only have an `Id` attribute if this is assigned in the election system after the votes are cast.

This schema allows for Simulation and Extrapolation of Results using data from Counts. Simulation being the facility to forecast the result of a contest based on the result of another contest. Extrapolation is the facility to forecast the final result of a contest based on the count so far.

Statistics (530)

This schema allows for a variety of statistical information to be made available about the counts and results captured in the Counts 510 schema. For example statistics about attendance and votes at each district and county level or by which voting channels have been used.

The statistics can be made available through any type of outlet be it Web, TV, SMS etc. and to any type of organization eg news agencies, political parties.

Options Nomination (610)

This schema is used to submit proposals, for example for a referendum or company AGM. It uses the generic Proposal element to define the proposal itself. One of more proposers can be named and may sign the nomination.

Options Nomination Response (620)

This message is sent from the election organiser to the proposer to say whether the nomination has been accepted. Along with the acceptance information and the basic information of election, contest and identifier for the proposal, a remark can be made explaining the decision.

Additional Rules

Error Code	Error Description
3620-001	If the nomination has not been accepted, a reason for rejection is required in the `Remark` element

Options List (630)

This schema is used for messages transferring lists of proposals for a referendum. It may identify the election event, and provides details about the election. Each proposal in a referendum counts as an election, so each election identified will hold a single proposal.

6. Conformance

To conform to this specification, a system must implement all parts of this specification that are relevant to the interfaces for which conformance is claimed. The required schema set will normally be part of the conformance criteria and should indicate schema version numbers. For example, the specification for an election list system might specify that a conforming system must accept and generate XML messages conforming to the following illustrative capability matrix:

Schema	Accept	Generate
EML110	V5.0, V6.0	
EML310	V5.0, V6.0	
EML330		V6.0
EML340		V6.0
EML350		V6.0
EML360		V6.0

A conforming system will then conform to the relevant parts of the EML specification and the accompanying schemas.

A. Acknowledgements

The following individuals in addition to the editors have participated in the creation of this Specification and are gratefully acknowledged:

Participants:

Siobhan Donaghy, OPT2VOTE Ltd
Bruce Elton, Oracle Corporation
Joseph Hall, University of California, Berkeley
Roy Hill, OPT2VOTE Ltd
John Ross, Secstan
Paul Spencer, Boynings Consulting Ltd
Bernard Van Acke, IBM
Sven Rubben, IBM
Peter Zelechoski, ES&S
Richard Cardone, IBM
Jeffery O'Neill, Individual

B. Other Considerations

B.1 Security

This section presents a general discussion of many of the security considerations commonly found in many election environments. As presented previously, these standards apply at EML interface points and define data security mechanisms at such interface points. This document is not intended to provide a complete description, nor a set of requirements for, secure election systems. In fact, the data security mechanisms described in this document are all optional, enabling compliance with these standards without regard for system security at all.

This discussion is included here simply to show how the information passed through the various interfaces described in these standards could be secured and used to help meet some of the requirements commonly found in some elections scenarios.

Basic Security Requirements

The security governing an election starts before the actual vote casting. It is not only a matter of securing the location where the votes are stored. An intensive analysis into security related concerns and possible threats that could in one way or another affect the election event resulted in the following:

- Security considerations of e-voting systems include:
- Authentication
- Privacy/Confidentiality
- Integrity
- Non-repudiation

Authentication

This is checking the truth of a claim of identity or right to vote. It aims to answer questions such as “Who are you and do you have the right to vote?”

There are two aspects of authentication in e-voting systems:

- Checking a claim of identity
- Checking a right to vote.

In some e-voting scenarios the two aspects of authentication, checking a claim of identity and checking a right to vote, may be closely linked. Having checked the identity of the voter, a list of authorized voters may be used to check the right to vote.

In other scenarios the voter’s identity must remain private and must not be revealed by a ballot. In which case some systems may provide a clear separation between checking of the claim of identity, which may be done some time before the ballot takes place, from checking the right to vote at the time of the vote is cast. Alternatively, other mechanism may be used to ensure the privacy of the voter’s identity on cast votes (i.e. by anonymizing the ballot).

In the physical voting world, authentication of identity is made by using verifiable characteristics of the voter like handwritten signatures, address, etc and physical evidence like physical IDs; driver’s license, employee ID, Passport etc, all of this can be termed a physical ‘credential’. This is often done at the time an electoral register is set up, which can be well before the actual ballot takes place.

Checking the authenticity of the right to vote may be performed at various stages in the process. Initial authenticity checks may be done related to the voter’s identity during registration.

Where an election scenario demands anonymity of the voter and privacy of the voter's ballot, the identity of the voter and the cast votes must be separated at some time within the voting process. This can be done in several ways by a voting system including, but not restricted to, the following options:

Authentication of the right to vote by itself does not reveal a voter's identity, but does verify he has a legitimate right to vote (e.g. the VToken data provides authentication of the right to vote but has anonymous properties as to the identification of the person voting).

An voter's identity and the right to vote are both validated (i.e. the VToken data has both 'voter identification' and 'right to vote' authentication properties) and then the cast votes are clearly separated from the identity of the voter (i.e. the voters identification occurs before the ballot is 'anonymized')

In all cases any verification of the authenticity that takes place after the voter has indicated his/her choices must preserve the privacy of those choices according to the laws of the jurisdiction and the election rules.

Finally, when counting and auditing votes it is necessary to be able to check that the votes were placed by those whose right to vote has been authenticated.

Public democratic elections in particular will place specific demands on the trust and quality of the authentication data. Because of this and because different implementations will use different mechanisms to provide the voter credential, precise mechanisms are outside the scope of this document.

Privacy/Confidentiality

This is concerned with ensuring information about voters and how votes are cast is not revealed except as necessary to count and audit the votes. In most cases, it must not be possible to find out how a particular voter voted. Also, before an election is completed, it should not be possible to obtain a count of how votes are being cast.

Where the user is remote from the voting system then there is a danger of voting information being revealed to someone listening in to the communications. This is commonly stopped by encrypting data as it passes over the communications network.

The other major threat to the confidentiality of votes is within the system that is collecting votes. It should not be possible for malicious software that can collect votes to infiltrate the voting system. Risks of malicious software may be reduced by physical controls, careful audit of the system operation and other means of protecting the voting systems.

Furthermore, the results of voting should not be accessible until the election is complete. Potential approaches to meeting this goal might include access control mechanisms, very careful procedural control over the voting system, and various methods of protecting the election data using encryption techniques.

Integrity

This is concerned with ensuring that ballot options and votes are correct and unaltered. Having established the choices within a particular ballot and the voter community to which these choices apply, the correct ballot information must be presented to each voter. Also, when a vote is placed it is important that the vote is kept correctly until required for counting and auditing purposes.

Using authentication check codes on information being sent to and from a remote voter's terminal over a communications network generally protects against attacks on the integrity of ballot information and votes. Integrity of the ballot and voting information held within computer systems may be protected to a degree by physical controls and careful audit of the system operation. However, much greater confidence in the integrity of voting information can be achieved by using digital signatures or some similar cryptographic protection to "seal" the data.

The fundamental challenge to be met is one of maintaining voter privacy and maintaining the integrity of the ballot.

Non-Repudiation

Non-repudiation is a derivative of the identification problem. Identification in e-voting requires that the system provide some level of assurance that the persons representing themselves as valid participants (voters, election workers, etc.) are, in fact, who they claim to be. Non-repudiation requires that the system provides some level of assurance that the identified participant is not able to successfully assert that the actions attributed to them via the identification mechanism were, in fact, performed by someone else. The two requirements are related in that a system with a perfect identification mechanism and undisputable proof of all actions would leave no room for successful repudiation claims.

Non-repudiation also requires that the system provide assurance that data or actions properly associated with an identified participant can be shown to have remained unaltered once submitted or performed. For example, approved candidate lists should be verified as having come from an authorized election worker, and voted ballots from a valid voter. In both cases the system should also provide a way to ensure that the data has remained unchanged since the participant prepared it.

Non-repudiation is not only a technical quality of the system. It also requires a certain amount of pure policy, depending on the technology selected. For example, in a digital signature environment, signed data can be very reliably attributed to the holder of the private key(s), and can be shown to be subsequently unmodified. The policy behind the acceptance of these properties, however, must be very clear about the responsibilities of the private key holders and the required procedures for reporting lost or stolen private keys. Further, and especially in “mixed-mode” elections (where voters can chose between multiple methods of voting), it may often be desirable to introduce trusted time stamps into the election data stream, which could be used to help determine acceptance criteria between ballots, or help resolve issues with respect to the relative occurrence of particular events (e.g. ballot cast and lost keys reported). The presence of the time information itself would not necessarily enable automatic resolution of these types of issues, but by providing a clear ordering of events could provide data that can be fed into decisions to be made according to established election policy.

Terms

The following security terms are used in this document:

- Identity Authentication: the means by which a voter registration system checks the validity of the claimed identity.
- Right to vote authentication: the means by which the voting system checks the validity of a voter’s right to vote.
- VToken: the means by which a voter proves to an e-voting system that he/she has the right to vote in a contest.
- VToken Qualified: the means by which a VToken can be qualified. The reason for the qualification is always appended to a VToken that is qualified. For example, a qualified VToken may be issued to a challenged voter.
- Vote sealing: the means by which the integrity of voting data (ballot choices, vote cast against a given VToken) can be protected (e.g. using a digital signature or other authentication code) so that it can be proved that a voter’s authentication and one or more votes are related.

Specific Security Requirements

Electronic voting systems have some very specific security requirements that include:

- Only legitimate voters are allowed to vote (i.e. voters must be authenticated as having the right to cast a vote)
- Only one set of choices is allowed per voter, per contest
- The vote cannot be altered from the voter’s intention
- The vote may not be observed until the proper time

- The voting system must be accountable and auditable
- Information used to authenticate the voter or his/her right to vote should be protected against misuse (e.g. passwords should be protected from copying)
- Voter privacy must be maintained according to the laws of the election jurisdiction. (Legal requirements of public elections in various countries conflict. Some countries require that the vote cannot be tracked back to the voter's identity, while others mandate that it must be possible to track every vote to a legitimate voter's identity)
- The casting options available to the voter must be genuine
- Proof that all genuine votes have been accurately counted.

There are some specific complications that arise with respect to security and electronic voting that include:

- Several technologies may be employed in the voting environment
- The voting environment may be made up of systems from multiple vendors
- A voter may have the option to vote through alternative delivery channels (i.e. physically presenting themselves at a poling station, by post, by electronic means)
- The voting systems need to be able to meet various national legal requirements and local voting rules for both private and public elections
- Need to verify that all votes are recorded properly without having access to the original input
- The mechanism used for voter authentication may vary depending on legal requirements of the contest, the voter registration and the e-voting systems for private and public elections
- The user may be voting from an insecure environment (e.g. a PC with no anti-virus checking or user access controls).

In addition, the objectives of security architectures for electronic voting systems should include:

- Being open
- Not restricting the authentication mechanisms provided by e-voting systems
- Specifying the security characteristic required of an implementation, allowing for freedom in its precise implementation.
- Providing the means to exercise security isolation and controls at interfaces between various election processes, thereby providing the ability to implement isolated trusted logic processes to meet dedicated functions of an election service. Process security isolation ensures that one voting sub-process does not inadvertently effect another voting sub-process thereby undermining the whole voting system.

Security Architecture

The architecture proposed here is designed to meet the security requirements and objectives detailed above, allowing for the security complications of e-voting systems listed.

The architecture is illustrated in figure 3a below, and consists of distinct areas:

- Voter identification and registration
- Right to vote authentication
- Protecting exchanges with remote voters
- Validating Right to Vote and contest vote sealing
- Vote confidentiality.
- Candidate list Integrity

- Vote counting accuracy
- Voting system security controls.

Voter identification and registration

The Voter identification and registration is used to identify an entity (e.g. person) for the purpose of registering the person has a right to vote in one or more contests, thus identifying legitimate voters. The security characteristics for voter identification are to be able to authenticate the identity of the legal person allowed to vote in a contest and to authenticate each person's voting rights. The precise method of voter identification is not defined here, as it will be specific to particular voting environments, and designed to meet specific legal requirements, private or public election and contest rules. The voter registration system may interact with the e-voting system and other systems to define how to authenticate a voter for a particular contest.

Voter identification and registration ensures that only legitimate voters are allowed to register for voting. Successful voter registration will eventually result in legitimate voters being given a means of proving their right to vote to the voting system in a contest. Depending on national requirements or specific voting rules/bylaws the voter may or may not need to be anonymous. If the voter is to be anonymous, then there must not be a way of identifying a person by the means used to authenticate a right to vote to the e-voting system. Right to vote authentication is the means of ensuring a person has the right to cast a vote, but it is not the identification of the person.

Right to vote authentication

Proof of the right to vote is done by means of the VToken, which is generated for the purpose of authentication that the voter has a legitimate right to vote in a particular contest.

The security characteristic of the VToken and hence its precise contents may vary depend on the precise requirements of a contest, the supplier of the voter registration system, the e-voting system, the voting channel or other parts of the electoral environment. Thus, the content of the VToken will vary to accommodate a range of authentication mechanisms that could be used, including; pin and password, encoded or cryptographic based password, hardware tokens, digital signatures, etc.

The contents of the VToken may also depend on the requirements of a particular contest, which may mandate a particular method be used to identify the person and the voter. For example, if a country has a national identity card system, it could be used for the dual purpose of identifying the person and providing proof that the person is entitled to vote, provided the legal system (or the voting rules of a private election) allow a personal identity to be associated with a vote. However, this would not work for countries or private voting scenarios that require the voter to be anonymous. For such a contest the mechanism used to identify that a person has the right to cast a vote must not reveal the identity of the actual person, thus under such voting rules voter identity authentication and right to vote authentication do not use the same information or semantics.

The security characteristic required of the VToken may also vary depending on legal requirements of a country or electoral rules used in a particular contest. Also, the threats to misuse of VTokens will depend to a large degree on the voting channels used (e.g. physical presence at voting station, Internet, mobile phone). Bearing this in mind the XML schema of the VToken components must allow for various data types of authentication information to be contained within it.

It must be possible to prove that a VToken is associated with a vote cast and the rules of the contest are followed, such as only one vote being allowed per voter, per contest. Thus providing proof /non-repudiation that all votes were genuine, they were cast in accordance with the rules of the contest, that no vote has been altered in any way and that all the votes counted in a contest were valid when audited.

Depending on the legal requirements of a country or electoral rules a voter may be challenged as to the right to vote, or may be given a temporary right to vote. In such cases the VToken may need to be qualified with a reason. In this document this is called a VToken Qualified. Before a vote is considered

legitimate and counted the reason for the qualification must have been suitably scrutinized, which could be done by the voting officials.

Protecting exchanges with remote voters

The VToken may be generated as part of the registration system, the e-voting system, or as interaction between various components of a voting environment, as illustrate in Figure 3a. The VToken will need to be provided securely to the voter so that this can be used to prove the right to vote.

The exchange of information when casting a vote must be protected by secure channels to ensure the confidentiality, integrity of voting data (VToken(s) and vote(s) cast) and that this is correctly delivered to the authenticated e-voting system. If the channel isn't inherently secure then this will require additional protection using other mechanisms. Possible mechanisms might include: a postal system with sealed envelopes, dedicated phone channel, secure e-mail, secure internet link (SSL), peer to peer server/client authentication and a seal.

Wherever technically possible the exchange of information should be secured and integrity guaranteed even if non-secure communications channels are used.

Validation right to vote and contest vote sealing

When a vote is cast, to ensure that it cannot be altered from the voter's intention, all the information used to authenticate the right to vote and define the vote cast must be sealed to ensure the integrity and non-repudiability of the vote. This seal may be implemented using several mechanisms ranging from digital signatures (XML and CMS), cryptographic seals, trusted timestamps and other undefined mechanisms. The seal provides the following security functions:

- The vote cannot be altered from the voter's intention
- The voting system is accountable and auditable.

The right to vote may be validated at the time the vote was cast. If votes are not checked for validity before sealing then the right to vote must be validated at the time that votes are subsequently counted. Also when counting, or otherwise checking votes, the validity of the seal must be checked.

If votes are sealed and recorded without being checked for validity at the time they were cast, then the time that the vote was cast must be included in the seal, so that they may be checked for validity before they are counted.

In some election scenarios it is required to audit a vote cast to a particular voter, in this case a record is also needed of the allocation of a VToken to a voter's identity. Such systems also provide non-repudiation of the voter's actions. In such cases a voter cannot claim to have not voted or to have voted a different way, or that his vote was not counted. In many election scenarios where this type of auditing is required, it must not be easy to associate a VToken to the Voter's identity, therefore this type of records must be under strict control and protected by security mechanism and procedures, such as; encryption, key escrow and security operating procedures.

Vote Confidentiality

All cast votes must not be observed until the proper time, this requires confidentiality of the vote over the voting period, how this is achieved will vary from e-voting system to e-voting system. Mechanism of vote confidentiality, range from trust in the e-voting systems internal security functions (processes and mechanisms) to encryption of the data, with key escrow tools.

Candidate List integrity

To ensure that the voter is present and that the candidate list is genuine, there must be a secure channel between the voting system and the person voting or the data must be sealed. The approach selected must ensure that there is no man-in-the-middle that can change a vote from what the voter intended. There are various ways this requirement can be met, ranging from the candidate list having unpredictable

characteristics with a trusted path to convey that information to the voter, to trust placed in the complete ballot/vote delivery channel.

As an example, there may be a secure path to convey the VToken to the person entitled to vote, a way of ensuring that a voter is always presented with a genuine list of candidates might be to encode the candidate list as part of a sealed VToken.

In summary, there must be a way of ensuring the validity of the ballot options and voter selection.

Vote counting accuracy

Audit of the system must be able to prove that all vote casts were genuine and that all genuine votes were included within the vote count. Voters may need to be able to exercise that proof should they so desire. Thus auditing needs data that has non-repudiation characteristics, such as the VToken/vote sealing, see schema 470 and 480.

Voting System Security

The overall operation of the voting systems and its physical environment must be secure. Appropriate procedural, physical and computing system controls must be in place to ensure that risks to the e-voting systems are met. There must be a documented security policy based upon a risk analysis, which identifies the security objectives and necessary security controls.

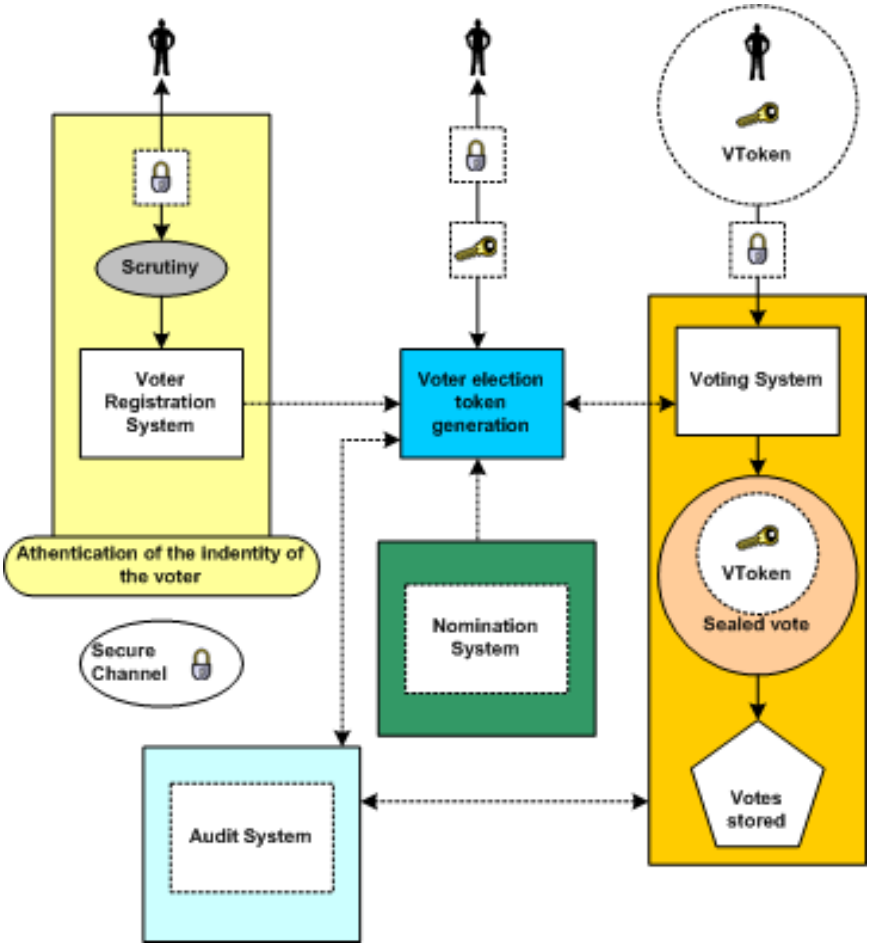


Figure 3A: Voting system security

Remote voting security concerns

Many new election systems are currently under evaluation. These systems tend to offer deployment options in which the communication between the voter and the election officials is carried out in an environment that is not completely under the control and monitoring of the election officials and/or election observers (e.g., the Internet, private network, telephones, cable TV networks, etc.). In these 'remote' or 'unattended' environments, several particular security concerns and questions like:

- How do I know that that the candidate information I am being presented with is the correct information?
- How do I know that my vote will be recorded properly?
- How do I know there isn't a man-in-the-middle who is going to alter my vote when I place it?
- How do I know that it is the genuine e-voting server I'm connected to that will record my vote rather than one impersonating it that's just going to throw my vote away?
- How do I know that some component of the system does not have malicious software which will attempt to alter the ballot choices as represented to me or alter my election?

The type and importance of a particular contest will have an effect on whether the above concerns exist and whether they do, or do not, represent a tangible threat to the voting process and its outcome. The table listed at Appendix B2 shows the concerns that have been identified as possibilities for one such remote or unattended environment (the Internet) that could be used in public election voting scenarios. The table shows how the concerns can be translated to technical threats and characterizes security services that may be used to counter such threats. Many of the items are not unique to the Internet, and can serve as a useful reference or starting point in developing similar threat analysis for other digital and/or unattended voting environments. How the security services are implemented in any particular environment or deployment is outside the scope of this document allowing freedom to the system providers.

B.2 Internet Voting Security Concerns

Concerns raised on Internet voting		Resulting Technical Threats	Possible generic security service countermeasure
1.	Impersonation of the right to vote. The concern here is that a person attempts to impersonate to be a legitimate voter when he/she is not. The initial task of verifying that a person has the right to vote must be part of the voter registration process.	Inadequate, incorrect or improper identification of person during registration of voters	Trusted voter identification and registration using: Security Procedures. Best Practices. Secure communications channels. The voter registration authority must follow standard Security Operating Procedures (SOPs) which ensure due diligence has been done.

	A person must not be given the right to vote until after proper due diligence has been undertaken during voter registration that the person has a right to vote in a contest.	Inadequate privacy of the exchange between the person and the electoral system during voter registration	Channel between voter and registration system must provide: Connection Confidentiality Connection Integrity
2	Voter is not presented with correct ballot information due to incorrect candidate identification.	Incorrect identification during candidate registration.	Trusted candidate identification and registration are needed using: - Security Procedures. - Best Practices. - Secure communications channels. - Authentication and identification of candidates The candidate registration must follow standard Security Operating Procedures (SOPs) which ensure due diligence has been done.
3	Registration system impersonation	Inadequate authentication of registration system	Channels to and from the registration system must provide point to point authentication.
4	Impersonation of a legitimate registered voter	Incorrect authentication at the time of casting vote.	Trusted voter authentication (i.e. the right to cast a vote in this contest)
		Inadequate privacy of the exchange between the voter and the electoral system when vote is cast.	Channel to provide: - Connection Confidentiality - Connection Integrity - Between voter and e-voting system

5	Obtaining the right to vote illegally from a legitimate voter. This may be by intimidation, theft or by any other means by which voting right has been obtained illegally. For example, by Stealing a voting card from a legitimate voter.	Stealing the voter's voting card (e.g. the VToken data).	Some secret data only known to the voter's is required to be presented at the time of casting a vote. Before a vote is counted as a valid vote proof must be provided that the voter's secret data was present at the time of casting the vote.
		Any means of getting a legitimate voter to reveal his VToken data.	
6	Voting system impersonation	Inadequate authentication of registration system	Channel to provide: Point to point authentication
		Inadequate authentication of voting casting point (e.g. polling station/ballot box)	Channel to provide: Point to point authentication
7	Voter is not presented with correct ballot information	Inadequate integrity of the ballot information Given to the user Held in the voting system	Trusted path to voter on ballot options
			Integrity of the ballot information
			Integrity of cast votes
		The casting options available to the voter are not genuine	Trusted path between voter and vote recording
		Trojan horse, man in the middle attack	Trusted path to voter on ballot options
8	How do I know the voting system records votes properly	Integrity of the voting system	Non-repudiation of the vote
			Non-repudiation the vote was cast by a genuine voter
			Audit of voting system
			Connection confidentiality
		Insecure channel between the voter and the vote casting point	Connection Integrity
			Connection Confidently
		Voter's intent is recorded accurately	Trusted path between voter and vote recording
			Non-repudiation of the vote

			recorded
		Proof that a genuine vote has been accurately counted	Audit
9	How can I be sure the voting system will not disclose whom I have voted for	Voter's identification is revealed	Voter's identification is anonymous
			Vote confidentiality
10	How can it be sure that my vote has been recorded	Loss of vote	Proof of vote submission
11	How can I be sure there is no man-in-the- middle that can alter my ballot	Vulnerable client environment; Trojan horses Virus	Physical security
			Procedural security
			Unpredictable Coded voting information
		Interception of communication	Integrity of communications channel between client and server system
12	All votes counted must be have been cast by a legitimate voter	Voter impersonation	Voter authentication
		Audit facility fails to provide adequate proof	Non-repudiation of the vote record
			Non-repudiation that legitimate voters have cast all votes.
		Breaking the vote counting mechanisms	Independent audit
13	Only one vote is allowed per voter, per contest	Voter impersonation at registration	User registration security Procedures Voter Identification
		Multiple registration applications	
		Multiple allocation of voters credentials	Voter authentication
14	The vote cannot be altered from the voter's intention	Vulnerable client environment; Trojan horses Virus	Trusted path from voter's intent to vote record
			Vote integrity
			Vote non-repudiation
15	The vote may not be observed until the proper time	Votes may be observed before the end of the contest	Voter confidentiality
16	The voting system must be accountable and auditable		Non-repudiation of vote data.
			Audit tools
17	Identification and authentication information to and from the voter must be privacy protected	Loss of privacy	Channel to provide: Connection Confidentiality

18	The voter's actual identity may need to be anonymous	Voter's identification is revealed Denial of service attack	Voter's identification is anonymous
19	Denied access to electronic voting station		This needs to be counted by engineering the system to provide survivability when under denial of service attack.

1365

1366 B.3 The Timestamp Schema

1367 Although used as part of EML, this schema has been put in a separate namespace as it is not an integral
1368 part of the language. A time-stamp binds a date and time to the sealed data. The time-stamp seal also
1369 protects the integrity of the data. The structure of the time-stamp is similar to the structure of an XML
1370 Signature.

1371 The timestamp structure may be used in one of two ways either:

1372 • Using Internet RFC 3161 binary encoded time-stamp token with the time-stamp information repeated
1373 in XML,

1374 • Using a pure XML encoded time-stamp.

1375 In the case of the RFC 3161 based time-stamp, the Timestamp structure is used as follows:

1376 • within TimestampedInfo:

1377 • TSTOrSignatureMethod identifies RFC 3161.

1378 • Reference contains the URI reference of the voting data being time-stamped. The DigestValue sub
1379 element contains the digest of the voting data being time-stamped.

1380 • TSTXMLInfoReference is not present in this case.

1381 • SignatureOrTSTValue holds the RFC 3161 time-stamp token applied to the digest of
1382 TimestampedInfo. The TimestampedInfo is transformed to a canonical form using the method
1383 identified in CanonicalizationMethod before the digest algorithm is applied.

1384 • KeyInfo contains any relevant certificate or key information.

1385 Object contains the TSTXMLInfo element which is a copy of the information in SignatureOrTSTValue
1386 converted from RFC 3161 to XML encoding. The TSTXMLInfo element contains:

1387 • the version of time-stamp token format. This would be set to version 1

1388 • the time-stamping policy applied by the authority issuing the time-stamp,

1389 • the time-stamp token serial number,

1390 • the time that the token was issued, the contents of this element indicate the time of the timestamp.

- optionally an indication as to whether the time-stamps are always issued in the order that requests are received
- optionally a nonce[2] given in the request for the time-stamp token,
- optionally the identity of the time-stamping authority

In the case of a pure XML encoded time-stamp, the Timestamp structure is used as follows:

- within TimestampedInfo,
- TSTOrSignatureMethod identifies the algorithm used to create the signature value.
- Reference contains the URI reference of the voting data being time-stamped. The DigestValue sub element contains the digest of the voting data being time-stamped.
- TSTXMLInfoReference must be present, and contains the URI reference of TSTXMLInfo as contained within the Object element. The DigestValue sub element contains the digest of the TSTXMLInfo.
- SignatureOrTSTValue contains the signature value calculated over the TimestampedInfo using the signature algorithm identified in TSTOrSignatureMethod having been transformed to a canonical form using the method identified in CanonicalizationMethod. This signature is created by the time-stamping authority.
- KeyInfo contains any relevant certificate or key information.

Object contains the XML encoded time-stamp information in an TSTXMLInfo element. The contents of TSTXMLInfo is the simular as for the case described above. However, in this case the information is directly signed by the time-stamping authority. The TSTXMLInfo element contains:

- version of time-stamp token format: This would be set to version 2
- the time-stamping policy applied by the authority issuing the time-stamp,
- the time-stamp token serial number,
- the time that the token was issued, this is the time of the timestamp.
- optionally an indication as to whether the time-stamps are always issued in the order that requests were received
- optionally a nonce given in the request for the time-stamp token,
- optionally the identity of the time-stamping authority.

B.4 W3C XML Digital Signature

Some information on the digital signature is included here, but for full information refer to the Recommendation at http://www.w3.org/TR/xmldsig-core/

An XML Signature consists of:

[2] A nonce is a parameter that varies over time and is used as a defence against a replay attack.

- SignedInfo which includes a sequence of references to the data being signed with the digest (eg. SHA-1 hash) of the data being signed
- SignatureValue which contains the signature value calculated over the SignedInfo using the signature algorithm identified in SignatureMethod having been transformed to a canonical form using the method identified in CanonicalizationMethod
- KeyInfo contains any relevant certificate or key information.
- Object can contain any other information relevant to the signature

C. Processing using Schematron or CAM

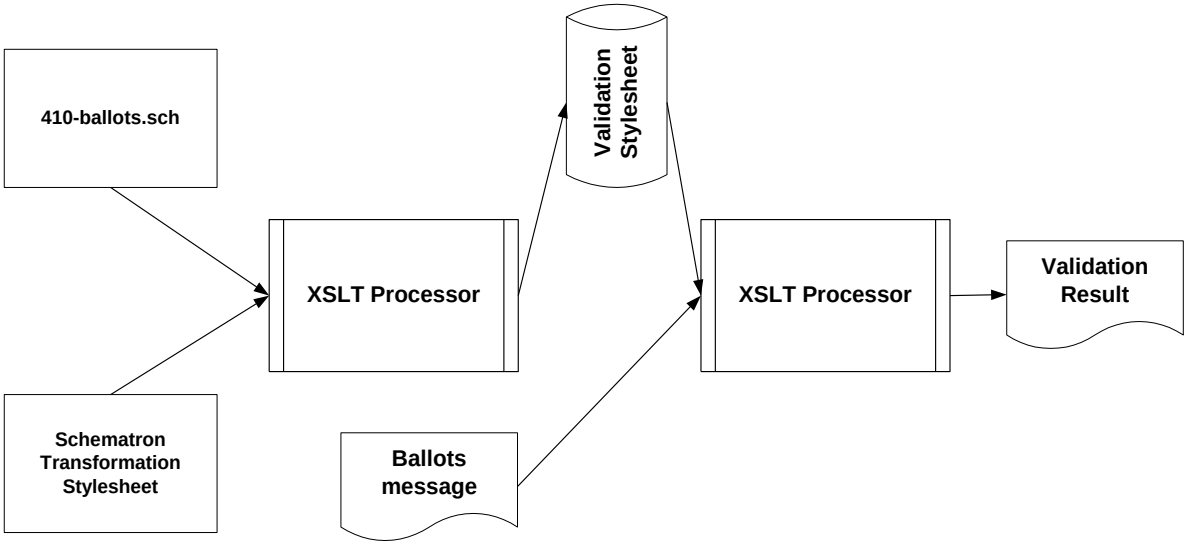
This section gives a short introduction to how validation can be achieved using either Schematron schemas or the OASIS CAM template approach. For Schematron this is done either using an XSLT processor tool (such as Saxon), or by direct validation using the Schematron schemas and a dedicated Schematron processor. For CAM templates this is using a conforming implementation toolkit such as the camprocessor project on SourceForge.net as open source.

Validation using Schematron Schemas

A Schematron schema is an XML document that can be converted to XSLT using an XSLT stylesheet. There is a published stylesheet (skeleton1-5.xslt) that can be used to achieve this. This produces an HTML output from the validation. A separate stylesheet can be produced that will create an output to the specification below. This stylesheet can import the skeleton and just over-ride those aspects where changes are required.

This stylesheet can be used once on each Schematron schema to produce the XSLT file that will be used for validating a specific message type. This stylesheet is then used to transform the incoming EML message into an error report based on the additional constraints.

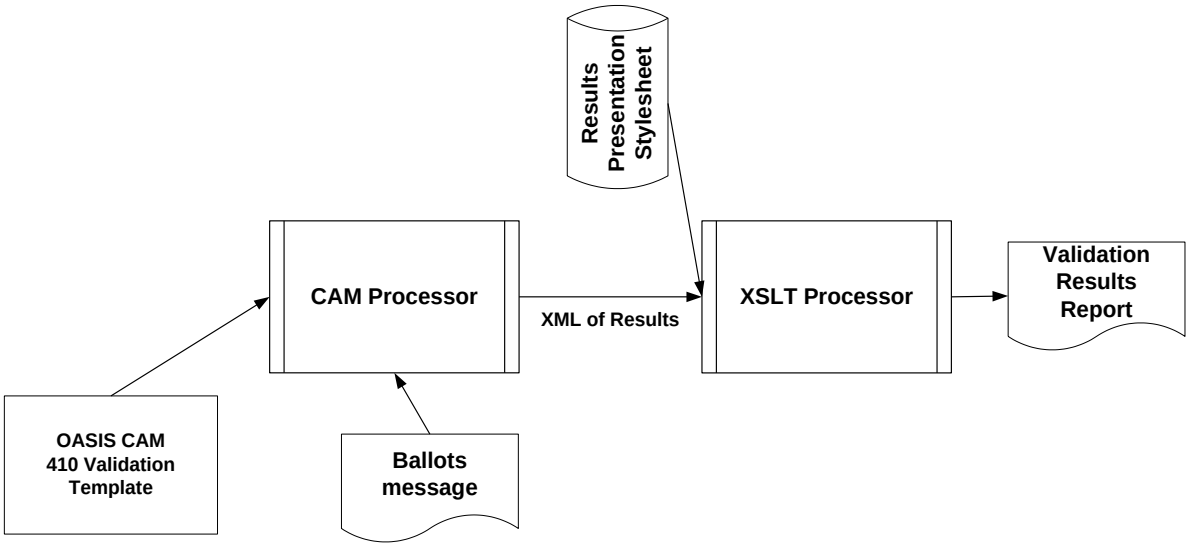
The process is shown in the diagram below.



Validation using OASIS CAM Templates

An OASIS CAM (Content Assembly Mechanism) Template is an XML document that provides the ability to rapidly tailor the XSD schema structure definitions in the base EML standard to suit country localizations and rules. The CAM template can then be used to validate the particular implementation XML transactions. An open source toolkit is available that implements the OASIS CAM specification. A default template can be generated using this toolkit by ingesting the particular EML XSD schema, and then tailoring that to produce a country localization pick list and customizations of the content rules. The toolkit will also allow the generation of realistic example XML test case instances and localization documentation.

1458 Once test cases and templates are available then these can be validated using the CAM toolkit. The
1459 process is shown in the diagram below.



1460
1461
1462

D. Revision History

Revision	Date	Editor(s)	Changes Made
V0.1a	2002-02-07	P Spencer	Draft e-voting schemas for internal comment
V0.2a	2002-02-13	P Spencer	Draft e-voting schemas for internal comment
V0.3a	2002-03-22	P Spencer	Draft e-voting schemas for public consultation comment
V0.4	2002-04-18	P Spencer	Draft Committee Specification version 2
V1.0	2002-04-29	P Spencer	Committee Specification for Technical Committee approval
V1.0	2002-05-13	P Spencer	Committee Specification
V2.0a	2002-06-13	F Ahmed	Revised draft accommodating committee's comments
V2.0b	2002-07-15	F Ahmed	Draft Committee Specification for Technical Committee approval
V2.0	2002-09-05	F Ahmed	Committee Specification
V3.0a	2002-12-12	F Ahmed	Draft Committee Specification
V3.0b	2003-02-06	F Ahmed	Draft Committee Specification for Technical Committee approval
V3.0	2003-02-24	F Ahmed	Committee Specification
V4.0a	2003-10-05	J Borras	Revised draft accommodating requirements of Council of Europe Member States and UK pilots
V4.0b	2004-01-27	J Borras	Draft Committee Specification
V4.0c	2004-03-09	J Borras	Revised draft by placing Schema Description section in document of its own due to excessive size of v4.0b. Draft Committee Specification for Technical Committee approval.
V4.0d	2004-09-03	J Borras	Draft Committee Specification for Technical Committee approval.
V4.0	2005-01-24	J Borras	Committee Specification
V4.0	2006-02-01	J Borras	OASIS Standard
V5.0	2007-03-14	J Borras	Committee Draft
V5.0	2007-09-12	J Borras	Committee Specification
V5.0	2007-12-01	J Borras	OASIS Standard
V6.0	2009-08-18	J Borras, D Webber	Committee Draft 01
V6.0	2009-08-18	J Borras, D Webber	Public Review Draft 01
V6.0	2010-06-11	J Borras, D Webber	Committee Draft 02
V6.0	2010-06-18	J Borras, D Webber	Public Review Draft 02

1465