Transparent Open Secure e-Voting using OASIS EML – Exploring the Paradox

Introduction

Citizens expect the voting process to be open and transparent and at the same time secure. When transferred to the digital domain these requirements set unique challenges for the implementing engineers. Usual software programming best practices and typical optimizations suddenly become significant and unintended security issues. In our approach using the OASIS Election Markup Language (EML), we explore how special care is taken to ensure that the e-Voting process is not inadvertently compromised. Particularly, the design focus taken ensures that the voting machines themselves have an anonymous view of the ballots they are handling and counting. That is, the program processing the ballot has no information about what any selected voter choice means; only the coordinates of the marked boxes is read. In this way, by limiting information from the scanning process, we seek to ensure that traditional concerns over voting attacks based on vote shifting and counting manipulation are ameliorated. During the voting, scanning and counting, none of the contest, candidate, and party or issue information is stored therein and hence this protects the programs from any external ability to shift results. Data on the meaning of voter marks is not anywhere on the computers involved in plain text or implied positional context; this presents a very significant hurdle for any hidden lines of rogue software to surmount. It also allows election officials to perform and verify the setup and operation of the ballots and counting independently of the software programmers. This transparency is vital in building trust.

Using fully documented open public specifications and storage formats, so all artifacts and components can be inspected and verified, creates a potential paradox; the precept of security through obscurity is violated. The approach, however, is central to creating a reliable shared software resource that can be re-used repeatedly for successive elections with simple procedures that do not involve programmers changing inner code. The core inner code is also very simple and brief so that practitioners can readily understand all aspects of the operations performed. Again this reinforces trust over time and use. Security is achieved through architecture rather than concealment.

A fully functioning open source implementation will be described here and samples illustrated. The support for and the mechanisms used to enable verification and auditing are shown and the challenges and paradox that this sets for potential attackers reviewed. Understanding these protection points allows physical procedures to reinforce the software safeguards.

The realization that, while you can never prevent new and innovative techniques that manipulate voting, you can set in place robust means to deter all but the most costly to develop in terms of resource. You can also elevate the detection risk to the point where more traditional non-computer based election manipulation techniques become the preferred means. This then achieves the objective of securing the ballot box process itself as the way for citizens to freely express their will by using a protected trusted environment.

What protects open and transparent designs?

The ability to verify the operation of the computer software and processes is crucial. Otherwise the voter is left with the sense of someone watching a magic trick on stage; that a sleight of hand is occurring, but it is impossible to know where. Similarly audit control and configuration control is vital for election officials to determine that they alone have control over the setup and operation of the ballot process without needing intervention from specialized programming staff.

Therefore our approach is one that exposes all aspects of the voting process in plain text working to open international standards. This means that every step can be directly inspected and understood; the artifacts produced known, and then verified. Similarly coding in the software programs is done in simple minimalistic modules that can be inspected and crosschecked. And each step is designed to perform just one simple and clear task. The goal is to minimize the opportunities for introducing rogue code. Also a secondary goal is to isolate the existing code from information that rogue code would require in order to manipulate the results.

Next even if rogue code does manage to be introduced, a further safeguard is provided by more than one set of records independently produced. This means that rogue code now needs a way to manipulate more than one set of records and communicate that across processes.
Now the other strong aspect of using open international standards is that an alternate code base may be used to test and confirm the counting itself by applying those common formats and methods. So one solution may be implemented in Java, and another in Ruby and both should produce the same results. Again this provides unique challenges for someone wanting to compromise both at the same time.

In fact we would argue that having an open and transparent process that can be simply and directly verified is the preferred protection mechanism. Using encryption and other complex obfuscation mechanisms merely removes the ability for operators to verify directly what is taking place using simple off-the-shelf common tools – such as text file viewers, browsers and editors or merely printing to a printer.

Securing the information trail

The traditional argument for encryption is to prevent tampering with records either during or after the physical storage to the recording media. Again this is a double-edged sword. Encryption itself implies decryption. Either the encryption or decryption process itself can be corrupted with rogue code that introduce hidden artifacts that simple tools cannot then see – and hence this could compromise the very use of open public storage formats and content specifications. Therefore we prefer to use content verification digital signatures within open text rather than encryption of the whole content.

Further in our approach printed paper records are used and digital images of those retained along with the matching XML record content. The intent is to provide three sets of independent records: simple XML content with a key value, a digital printer image that includes a barcode of the key value, and then a printed paper record of that which is used by the voter to cast their physical ballot. The digital printer image and the actual

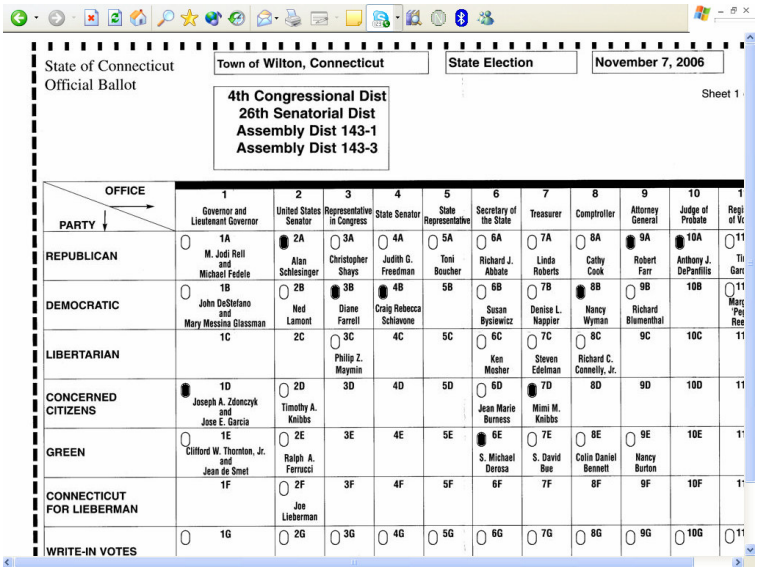
printing are controlled by a separate computer process to that which records the XML – this prevents a single computer being able to manipulate both together. And the only communication between the two is via the simple XML record. E.g. one obvious attack is to print a paper ballot with one key value but subsequently assign that to a different voter record. Again task division is the major protection. The second computer process does nothing more than read in the XML record and output the printer image and the paper ballot. The voter then directly verifies that the printed ballot matches the ballot choices and physically casts that in the ballot box. Subsequent auditing can then crosscheck all three components to confirm that they match.

Unified scanner/ballot handlers

Design simplicity and interchangeability is essential for a system that uses off-the-shelf hardware components and setup configurations. A physical voting system may contain only ballot scanners and counting systems, or voting terminals, scanners and counting, or any permutation of these three. Similarly the voting terminal configuration itself may be a laptop computer, a desktop computer, a thin-client terminal or even a cellular phone device customized for voting use. The common factor is simply that they have a display function along with an input keyboard or voice entry capability and the ability to securely run the voting software that works to the EML standards. In each case the output created is the simple XML content of the ballot record. And in each case the ballot selection display method is using minimal artifacts that do not expose the physical ballot choices only a simple coordinate system is employed. The Figures 1 and 2 here show how this works for a Connecticut State Election ballot.

Figure 1 is the physical view the voter sees – made up of a scanned GIF image of the entire paper ballot, overlaid with push-button controls that are positional – using row and column offsets – controlled by the XML shown in Figure 2 below.

Figure 1 – Example Ballot


State of Connecticut
Official Ballot

Town of Wilton, Connecticut | State Election | November 7, 2006

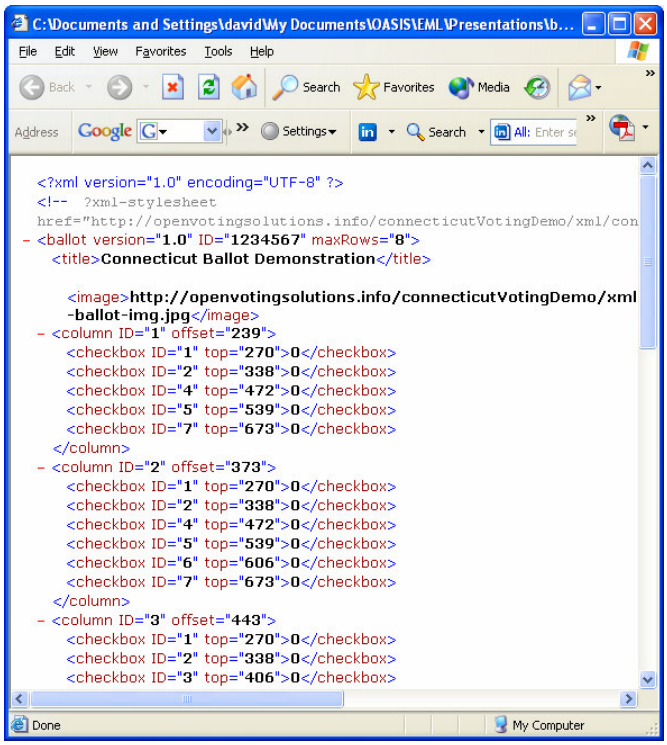
Sheet 1

4th Congressional Dist
26th Senatorial Dist
Assembly Dist 143-1
Assembly Dist 143-3

PARTY \ OFFICE	1 Governor and Lieutenant Governor	2 United States Senator	3 Representative in Congress	4 State Senator	5 State Representative	6 Secretary of the State	7 Treasurer	8 Comptroller	9 Attorney General	10 Judge of Probate	1
REPUBLICAN	1A M. Jodi Rell and Michael Fedele	2A Alan Schlesinger	3A Christopher Shays	4A Judith G. Freedman	5A Toni Boucher	6A Richard J. Abbate	7A Linda Roberts	8A Cathy Cook	9A Robert Farr	10A Anthony J. DePanfilis	11
DEMOCRATIC	1B John DeStefano and Mary Messina Glassman	2B Ned Lamont	3B Diane Farrell	4B Craig Rebecca Schiavone	5B	6B Susan Bysiewicz	7B Denise L. Nappier	8B Nancy Wyman	9B Richard Blumenthal	10B	11
LIBERTARIAN	1C	2C	3C Philip Z. Maymin	4C	5C	6C Ken Mosher	7C Steven Edelman	8C Richard C. Connelly, Jr.	9C	10C	11
CONCERNED CITIZENS	1D Joseph A. Zdonczyk and Jose E. Garcia	2D Timothy A. Knibbs	3D	4D	5D	6D Jean Marie Burness	7D Mimi M. Knibbs	8D	9D	10D	11
GREEN	1E Clifford W. Thornton, Jr. and Jean de Smet	2E Ralph A. Ferrucci	3E	4E	5E	6E S. Michael Derosa	7E S. David Bae	8E Colin Daniel Bennett	9E Nancy Burton	10E	11
CONNECTICUT FOR LIEBERMAN	1F	2F Joe Lieberman	3F	4F	5F	6F	7F	8F	9F	10F	11
WRITE-IN VOTES	1G	2G	3G	4G	5G	6G	7G	8G	9G	10G	11

This ballot can be completed with either a computer system entry (shown), or by physically marking a paper ballot with black pencil or pen. The actual details of the ballot are therefore not visible to the program that is controlling the display and the actions of the voter. Instead it has an anonymous view of the process as merely columns and rows of selections.

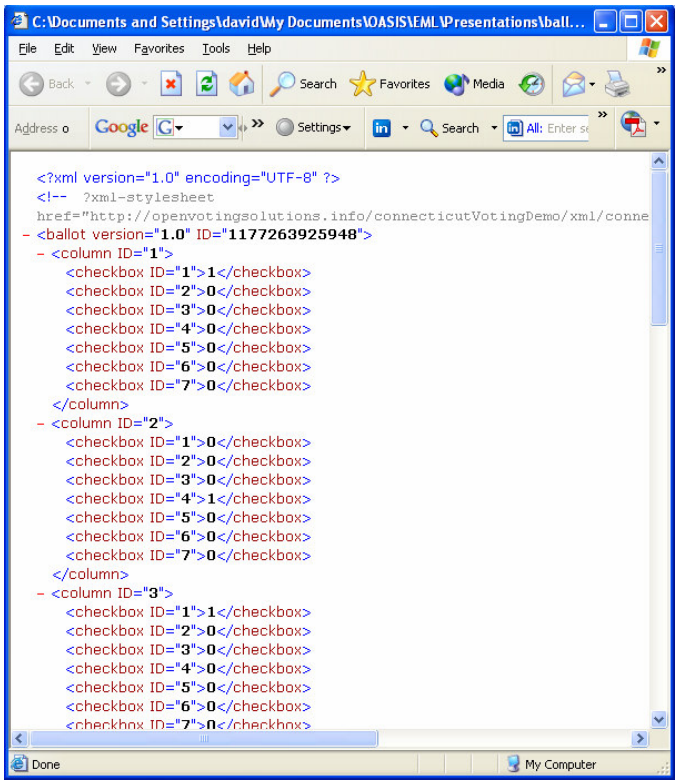
Figure 2 – XML layout control for ballot



Shown here are the row and control details and the layout positions (offsets) of the actual ballot entry checkbox buttons.

The output XML of the ballot choices is then stored in a simple result format as shown in Figure 3. This XML result format is identical whether created by a voting terminal device, or read by a scanner device from the physical paper ballot cast by the voter.

Figure 3 – Anonymous vote XML record



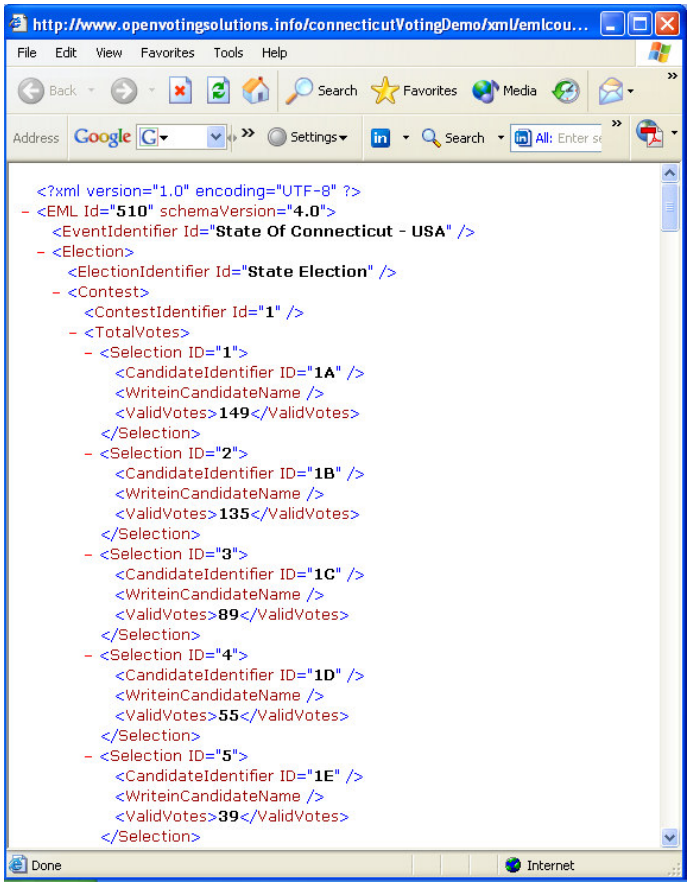
This record corresponds to the original layout definition XML. Notice the ballot record layout contains in addition the ID key value, and that choices are simply recorded as a value 1 and 0. It is critical that this vote XML information is anonymous – there are just checkbox id values – not actual textual details exposing the ballot choices. This also allows the one program code logic to handle any ballot format by simply changing the XML and no other logic in the processing itself. This includes automatically checking for under or voter votes or maximum number of selections within a selection column.

This simple XML is then also further stored in OASIS EML 440 ballot format, that includes additional audit and control information – such as the location and ballot station – and can also contain a digital signature line.

Anonymous counting protection

The EML 440 format is then used for anonymous counting purposes to ensure any standard counting component can confirm the result totals. All the counting totalling is performed at the level of the code values only. No candidate, party or issue textual information is exposed at this point. An anonymous lookup codelist created by election officials only can be further used to remove positional information. This produces EML 510 count total records that are anonymous. These totalling computations are then also performed separately from the ballot recording process either by the voting terminals or the scanner. Again it is vital that no totalling functionality be present during the initial vote processing.

Figure 5 – OASIS EML 510 totals



Then only once these totals are completed after all balloting is finished are the 510 results physically transferred to a separate reporting computer system. Only on this reporting system is the decoding needed to expose the coded choices into actual textual candidate and issue results. This decoding is provided by the OASIS EML 410 ballot layout format. By having a standard format for this critical information various other safeguards are provided. The EML 410 can be created by election staff, not by programmers alone. It can be verified to match the content rules for EML 410 to ensure no other information is there, or information in a format that is not permitted. It provides direct control to the election staff as the EML 410 can be kept sealed in a secure location until it is needed after all balloting is completed and totalling is finished.

The reporting system also performs other essential functions to allow auditing and verification of results by precinct, county and city areas that allow election monitoring staff to compare the digital results with reported exit polls and other physical sampling techniques.

Additional Physical Factors

While these digital techniques provide a unique set of tools for managing an election process it is also essential that staff involved in the election understand their own roles in securing what occurs. Too often we hear of vendor support staff performing last minute maintenance and fixes to election equipment and software immediately prior to voting. Clearly such unfettered tampering is what should be prevented at all costs. This is where the OASIS EML standards provide unique value. By allowing election officials to use standard software components that separate the operational process and ballot

information details it permits direct control over the operation and details and the software itself. This permits changes to be isolated and controlled.

Furthermore the OASIS EML is not just a software standard it also encompasses the voter bill of rights produced by the Council of Europe and endorsed by the council of ministers for the whole of the EU. As such is incorporates operational and procedural management and control details for election staff involved in the complete setup, running and then post-election auditing processes.

This is essential for the training and knowledge needed by election staff so that they understand each step, their role and actions permitted.

Reporting Results

The functions on the Election Day then complete with the results reporting. Even at this stage care must be taken to ensure that no manipulation is invited. The OASIS EML 520 provides the strict layout details for recording the results themselves. These are transferred from the EML 510 by looking up the actual candidate and issue information provided in the EML 410 ballot layout description. Then to prevent any option to manipulate the displaying of those results the W3C standard stylesheet programming language – xslt is used. This will take a given XML document and render it for viewing in a standard browser window at HTML. Again this removes the need to specific programmer coding of changed ballot and contest details. Everything is instead driven by the sealed copy of the OASIS EML 410 that the election officials retain strict control over.

Figure 6 – Results Reporting – OASIS 520

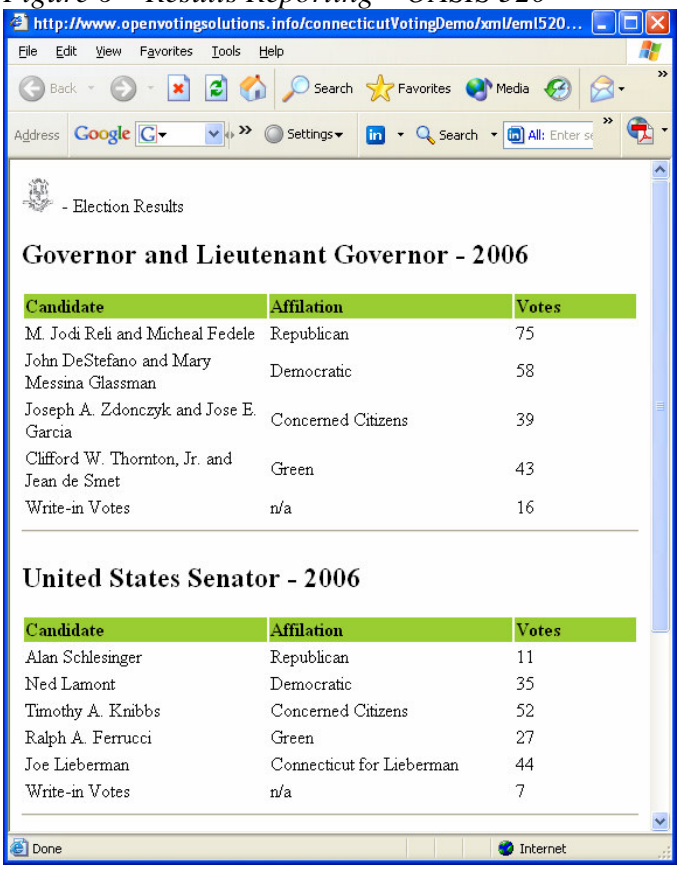
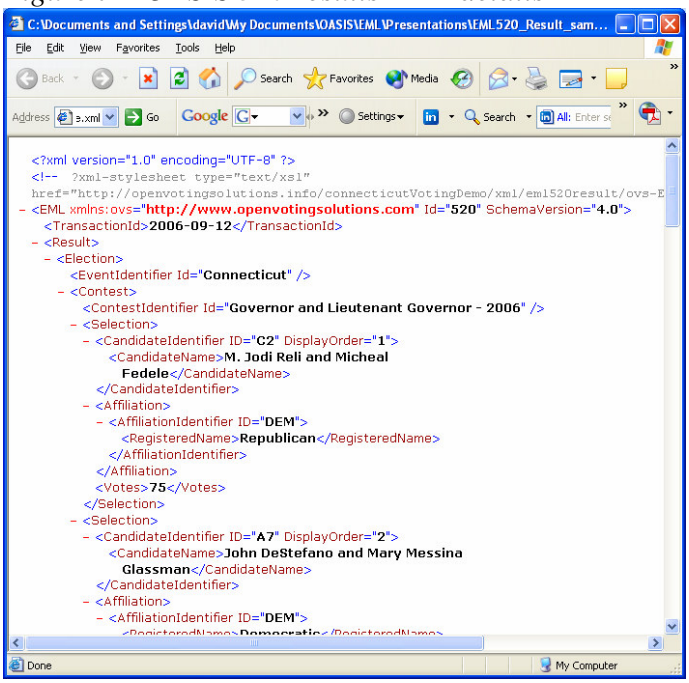


Figure 7 – OASIS 520 results XML details



Role of Open Source

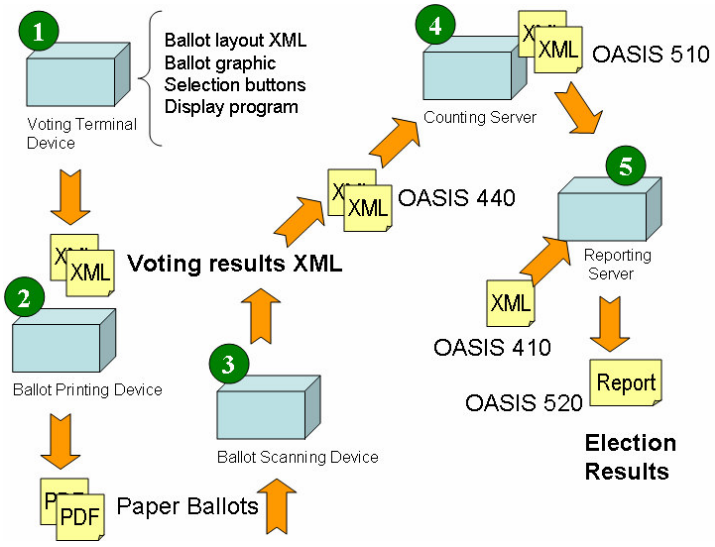
Associated with the OASIS EML XML processing is the use of open source components in our approach. Again the paradox is that software source code is fully exposed of all the high level components being used. Allied to this is the use of interchangeable hardware components that are off-the-shelf, not bespoke custom voting only products. The key word here is interchangeable. If it is suspected that the firmware or hardware itself is somehow harboring rogue code, then a change in behaviour should be detectable when compared to alternate hardware from a different supplier. Also it is then impossible for a supplier to know exactly where their equipment may be used, from election to election, and hence to be able to embedded direct election knowledge inside that. Especially given the safe guards taken to remove any textually or positional information in the software and XML formats used at the balloting sites.

Architecture of typical solution

The components described above compose a typical configuration that is illustrated in figure 8 here that conforms to the OASIS EML process specification. What the architecture does is separate out the components to support use of an anonymous counting approach and to allow ballot details to be controlled and configured by election staff.

This architecture is for illustration purposes only and actual delivery for an election will be determined by the local election legal and operational requirements.

Figure 8 – Solution Architecture



Conclusions

In summary the technical methods we have developed are based on the OASIS EML specifications and the use of open source and open standard technologies including XML and xslt from the W3C. The initial implementation is done using Java technology to ensure open platform deployment on any hardware supporting the Java environment.

The focus is on showing how the method and approach work and implementing a reference software solution that can then be purposed for actual election use. To develop a full election system from the prototype will require additional safeguards to be applied at the operating system level to ensure that only required components are present. Similarly the tailoring of the tools used to process and report the election ballots – most importantly the browser component need to be configured to prevent external tampering.

Other options include the use of "PollBook" voter registration systems (again OASIS EML compatible) along

with using physical balloting access devices – such as barcode "wands" - to permit authorized voters access only.

From all these base elements it is envisioned that a reference implementation can be finalized that will provide election authorities with the means to deploy their own solutions and to control those in a systematic way that is provided for by the OASIS EML process.

Going forward this can then provide the basis for a community supported open source resource for trusted elections using such digital counting systems for transparent secure e-voting.

While many see that there is a paradox in developing open transparent voting solutions we see this as a challenge that sound software engineering and design can overcome.

Resources

The OASIS EML specifications details: http://docs.oasis-open.org/elections
EML Open Source for voting software: http://emlvoting.org
The World Wide Web Consortium (W3C): http://www.w3c.org
The Case for OASIS EML white paper: http://www.oasis-open.org/committees/download.php/22101/The%20Case%20for%20EML.pdf
Background on OASIS EML XML: http://xml.coverpages.org/eml.html

Presenters:

- David RR Webber,
  - CTO Open Voting Solutions Inc.
- John Borras,
  - Chair of the OASIS EML TC
- Richard C. Johnson PhD
  - CEO Open Voting solutions Inc.

Biographies:

John Borras –

Formally Chief Executive of the UK Local e-Government Standards Body (LeGSB), supporting local government meet their e-service delivery targets. Prior to that John was the Director of Technology responsible for the UK's e-Government Interoperability Framework (e-GIF) and other e-government and e-voting technical policies and standards.

Richard C. Johnson, Ph.D. -
Formerly a Director of Development for Oracle Corp specializing in secure application area solutions, including distributed voting solutions. US patent holder and contributor to open source initiatives on voting. Over 25 years of industry experience.

David RR Webber, B.Sc. -
Industry recognized expert in application of XML to e-Business. Member of the OASIS EML TC standards work and a US patent holder for XML related technologies. Formerly VP Business Development with XML Global Technologies with over 25 years of industry experience.

Submission to EVT07 – August 6–10, 2007, Boston, MA

Applicable Topic Areas:

- Design and analysis of electronic voting schemes and protocols
- Deployment and lifecycle concerns
- Mitigating threats (including insider threats)
- The technology standards process and how it should evolve

Contact author:

David RR Webber,
david@openvotingsolutions.com
Tel: 301.693.1000

9005 Magruder Knolls Court,
Gaithersburg, MD 20882