



# TrustTheVote Data Layer

The Open Source Digital Voting (OSDV) Foundation  
TrustTheVote Project

R. Pito Salas, Core Team  
November, 2009

## 1. Background

In this paper I will introduce the TrustTheVote.org Data Layer, explaining what exactly we mean by that term, and it might evolve over time. The TTV project envisions the design and development of a complete suite of software and hardware to support the election process from voter registration all the way to the reporting and auditing of election results, and every step in between (I will refer to it as the TTV Suite from here on.)

In a later section of this paper I will present the specific components as currently envisioned. However we expect that as we move forward, as specific designs, names and scope of components become clear, things will shift. Yet the Data Layer as introduced in this paper should continue to be applicable.

### 1.1. Purpose

The Data Layer ensures that the TTV Suite is not an impregnable monolith or silo of proprietary data. It is made real by a set of commands, APIs, data formats, validators, converters and other technical components which together permit the open, transparent and flexible exchange of data.

The Data Layer enables what we describe of openness of data. In other words, in addition to having a clear focus on open-source technology, the TTV suite will employ open data. Open data, enabled by the TTV Data Layer, means:

- **Non-proprietary formats.** Other applications will be able to both read and write TTV data, whether it is to interface existing systems, or to audit the workings of the TTV system, or just to increase confidence and trust.
- **Readable text formats:** Data that is interchanged between components of the TTV suite will be formatted as XML, readable by people using simple text editor.

### 1.2. Relationship of components

As mentioned above, the TTV Suite is built from a series of relatively separate components, mostly software. These are all the more or less well known building blocks of election technology, for example:

- Election management System (EMS), to manage the data for each election, offices, districts, candidates, ballots, and so on.
- Ballot Design Studio (BDS), which will use data from the EMS to assist in the design, templating, and preparation for printing of sets of ballots.
- Ballot Counter / Scanner (BCS), to actually scan and count paper ballots.

As currently envisioned, the components are build either as data base driven *web based applications*, running on servers, accessed through web browsers, or as *stand-alone devices*, running our software on hardware specified as part of the TTV Suite..

The *web based applications* will sit atop databases and will often pass information between them simply by storing and reading from a shared database. In other scenarios they will exchange information via web based file transfer or email, or so-called Web Services.

The *stand-alone devices*, because they are not connected to a network, perforce will exchange information via writable media such as secure memory sticks or write-once DVDs.

In all these scenarios, still, the Data Layer defines the formats, protocols, and operations that support these various kinds of information exchange. Importantly, there are no restrictions on additional ways of exchanging information between components, most importantly via shared databases, as long as the mandatory Data Layer formats and processes are also fully supported.

### 1.3. Media and Files

As you can see, there are a variety of media that can be used to exchange information amongst the components. First of all there will be simple files interchanged on writeable media such as memory sticks or writeable cd-roms. Certain kinds of data might also be transferred as email attachments where appropriate. And in addition, the technology suite will support REST and other APIs for extensibility.

For the purposes of this document I will refer to 'data files' in each of these scenarios, calling out any special cases if and when appropriate.

### 1.4. XML

XML is the underlying format for all files. XML has well known advantages, or example it is considered more or less readable (although sometimes a challenge) by eye with a simple text editor, on any platform. It is also of course very well known and accepted in industry. And last but not least, it is very well supported by a spectrum of programming languages and platforms, providing great support for international character sets as well. The TTV suite of technology will support other formats such as YAML and JSON in other places where it makes sense.

## 1.5. Testing

In addition to writing code and specifications for working with the different data files, we will be paying careful attention to the creation of automated unit testing and file validation suites to verify files generated and consumed by our components as well as components added in the future.

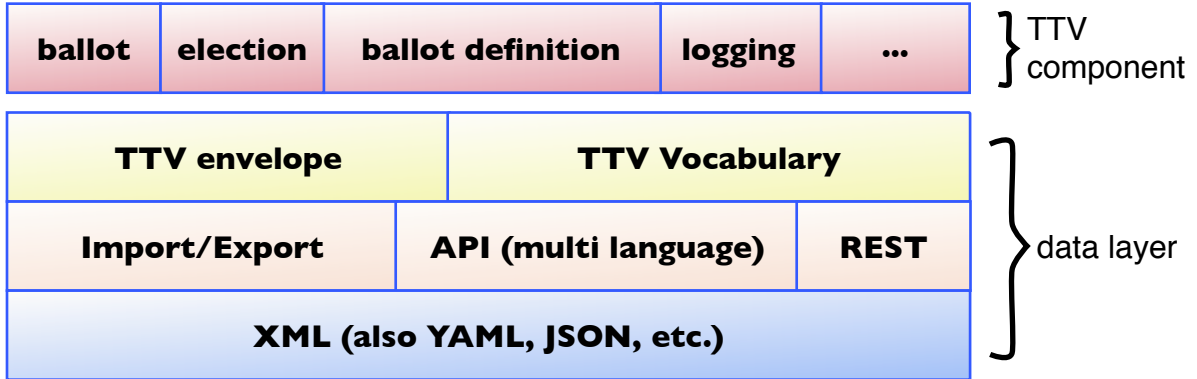
## 1.6. Logging

All TrustTheVote Suite components will pay careful attention to logging. It is critical to plan for this from the outset, so that it will be possible to reconstruct what happened during use of the components as well as back-tracing any problems, errors or alerts that might have occurred.

The Data Layer provides facilities and defines formats for standardized logging. In some scenarios, the Data Layer will itself generate log records, but more commonly this will be the responsibility of individual components.

Auditability of our components is a priority and as such is equally represented as a peer in the data layer.

## 2. Data Layer Architecture



As mentioned above, the TTV suite of technologies consist of a collection of separate components, some of which run on stand-alone devices (for example the Ballot Scanner and Counter) and some which run as web applications (for example, the Election Management System.)

The Data Layer's job is to allow each of these components to pass information and control in such a way as to ensure that the system never becomes a silo or monolith. In order to do this it must:

- Permit components of the suite to be used individually without requiring all of them to be implemented at once.
- Permit existing systems to connect up to newly deployed TTV components, allowing data to be sent to, from or between the systems.
- Permit information to be examined, interpreted and audited at any interface between components.

### 2.1. Domain Models sub-layer

This layer, the top one in illustration above, shown in red, will be implemented as part of each TTV Suite Component. That is to say, each component will be required to follow certain standard protocols to interface to the Data Layer.

Even though the file formats are the 'information of record', they need to be brought to life as memory-based objects, representing what are known as the domain models, objects for ballots, voters, districts, depending on the TTV Component.

So, for example, in the Election Management component, there will be Domain Models for election, precinct, district, and many more.

This sub-layer consists of implementations of these domain models. They will both represent the objects and their behavior (as Object Oriented Classes.) They will be implemented as part of individual TTV components, and wherever possible actually shared among components.

## **2.2. Standard File Content sub-layer**

The Standard File Content sublayer, illustrated above in green, standardizes the format of the disparate Domain Models into a consistent form. This is done as part of the process of turning a Domain Model class into a ‘file’ (for example to send information about a the result of a tabulation from the from a ballot scanner and counter to the tabulator.)

Every file is known by it’s file-type. A file-type might be something like “ballotinfo-tabulation-result.” All files include some elements irrespective of what kind of file it is, as well as other elements which are there by virtue of the file-type. The standard information is part of the “TTV Envelope” (see diagram above.)

The TTV Envelope includes a so-called “Unique ID” which uniquely identifies that specific file from all the others generated in this or any other election. It is a 128 bit number. It has nothing to do with encryption but is used to track the provenance of files. Other standard information might have to do with what device or application, and what version, generated the file, what files it was based on (if the result of some transformation.)

The rest of the information is structured according to the file-type. While the individual definitions are outside of the scope of this document, we can say that they will be based on a standard TTV ‘vocabulary’ which in turn will respect corresponding vocabulary in accepted standards such as EMS.

## **2.3. Data Interchange sub-layer**

The next layer, illustrated in the diagram above in pink establishes mechanisms for actually performing data interchange.

Every component in the TTV suite will include a data import/export feature. This feature will allow the state, either complete or partial, of the entities controlled by that component to be interchanged.

Even when the default manner for interchange is through a database, the option to accept input via an import (for example, from a system that pre-exists the installation of the TTV suite) or an export (for example, to a custom made reporting package). The import and export will accept and generate files.

In addition to the mandatory import/export feature the TTV suite will include APIs automate the interpretation of these files, thereby saving duplication of code and bugs, and making it easier for interfacing applications to be created.

And finally, as appropriate, there will be HTTP/REST based interfaces to further permit interconnection via web services.

#### **2.4. File Format sub-layer**

Finally, the file File Format sub-layer, illustrated in the diagram above in blue, has simply to do with the detail formatting of files. As mentioned above, all 'files' - meaning all self-contained blocks of information - transmitted between the components will be formatted in standard XML. There will be options for using related formats if desired, but XML support will be the default.

### **3. Conclusion**

This white paper introduced the approach that we will be taking in creating an open, transparent, testable and trustworthy data layer for the TrustTheVote election technology suite.