ErgoGroup

E-vote 2011
System Architecture
Overview, Interfaces and Deployment

V 1.5

Change history

Date	Version	Description	Author
07.12.2010	1.0	Initial	
08.12.2010	1.1	Adapted minor changes to area-code in electoral roll export.	
11.03.2011	1.3	Updates and minor corrections.	
15.03.2011	1.4	Added section on usage of virtual counting districts to preserve anonymity.	
25.04.2011	1.5	Added appendix with more details on how EML-config files is to be understood.	

Contents

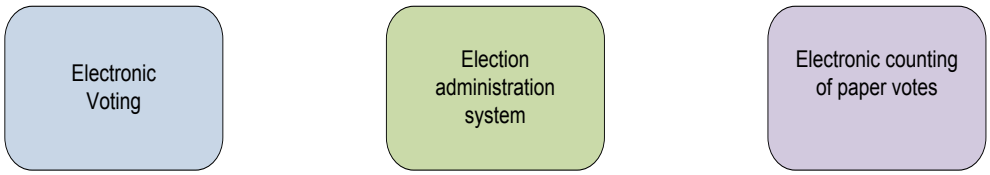
1. Introduction 4
   1.1 Definitions 4
2. Overall conceptual drawing 5
   2.1 Description 5
3. Interfaces between main system components 7
   3.1 Distribution of Election configuration 7
   3.2 List proposals 7
   3.3 Distribution of Electoral roll 8
   3.4 Upload of counts 9
4. File transfer and formats between main system components 10
   4.1 General on Signing 10
   4.2 Message Formats 10
   4.3 File transfer of election configuration 11
   4.4 File transfer of counting results 11
   4.5 Transfer of electoral roll, optionally with mark-offs, from Admin to eVote 12
   4.6 Usage of virtual voting districts to preserve anonymity 13
5. Deployment 14
   5.1 Infrastructure overview eVoting and Administration system 14
   5.2 Infrastructure overview eCounting system 19
   5.3 Securing connectivity between systems 21

1. Introduction

The purpose of the architecture documentation is to provide an architectural overview of the e-Vote system for

- Downstream designers and implementers
- Testers and integrators
- For those evaluating the system's quality objectives
- Technical managers and project managers
- Designers of other systems that need to interface with the e-Vote system

The E-Vote 2011solution has three different but interacting systems.

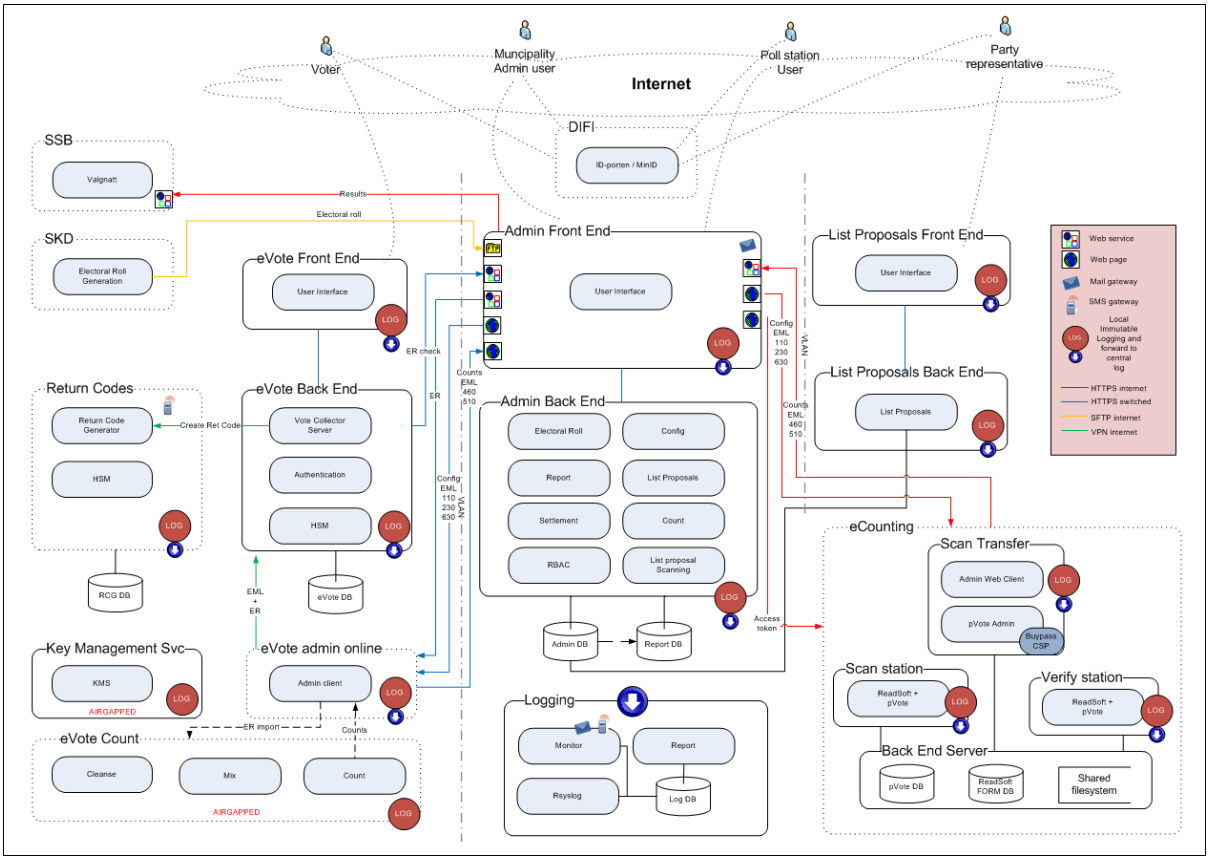


Each system can operate in its own environment, but require interaction to satisfy common requirements for configuration, authentication, reporting, auditing, and key management.

1.1 Definitions

E-vote 2011	The project
E-voting client	Subsystem for casting votes electronically
E-voting system (eVoting)	The subsystem handling eVoting. (Includes E-voting client, backend functions for storing votes, and functions for tallying votes i.e. counting of eVotes).
Election administration system (Admin)	Subsystem for configuring and managing elections
List proposals	Part of the Admin system
Electoral Roll (ER)	Part of the Admin system
Election configuration	Part of the Admin system
Counting registration	Part of the Admin system
Settlement	Part of the Admin system
eCounting of pVotes system (Scanning)	Subsystem for counting of paper ballots
Reporting	Horizontal reporting functionality
Log & Audit	Horizontal log and audit functionality
SKD	Norwegian Revenue Service (Source for ER)
SSB	Central Statistics Bureau (Receiver of election statistics)

2. Overall conceptual drawing



2.1 Description

The three systems have several different user groups. The main user groups are:

1. **Election administrators/officials** (centrally and at the municipalities) access the system to perform various tasks: election configuration, update electoral roll, run reports, approve list proposals from parties, count elections and perform settlement of elections. Selected users have access to add/change roles and users in the system. Officials at the poll stations use the system to verify voter eligibility in the electoral roll and to mark-off votes in the electoral roll.
2. **Political party officials** use a public web application in advance of the elections to submit party and candidate proposals.
3. **Eligible voters** cast their electronic vote in a public web application.
4. **Operators in eCounting of pVotes** scan paper ballots, verify them, count them, and report the results.
5. **Operators in eVoting** setup and configure the election and report the counting results from the electronic voting.

For authentication purposes, all systems integrate towards ID-porten (the national provider of electronic IDs) for central authentication. This is in principle a single-sign-on system for many government services, but the configuration does not allow single-sign on towards the election systems.

The electronic voting system (eVote) consists of the modules necessary for a voter to submit votes and receive return codes to their mobile phone that confirm their vote. The voting system has a front-end and a back-end for added security. In addition, the eVote system has its own administration module used by central officials to configure the behavior of the eVote client.

Counting of eVotes is performed in a separate airgapped environment and consists of three operations: “Cleanse” to remove ineligible votes, “Mix” to separate the voter from the vote, and “Count” to count the

electronic votes. After the election, the counts are signed and handed off to the central election system.

The administration system has all the modules necessary to manage an election: the electoral roll, election configuration, reporting, and a module for approving list proposals (party lists and candidates). Counting module that accepts manual counts and for verification of counts from other systems (eVote and eCounting). In addition, there is a module to perform settlement of an election (distribute seats). There are also two supporting modules, one for managing role based access and one for OCR interpretation of party list proposals.

The administration system has a front-end and a back-end for added security. As list proposals are made by the parties, there is a separate system to receive list proposals with its own front-end and back-end.

The administration system has two external interfaces beyond DIFI. It is integrated towards the Norwegian Revenue Service (SKD) such that daily updates to the Electoral Roll can be received. In addition there is an interface to send statistics (counts) to the Central Bureau of Statistics (SSB).

To support secure communication and signing, there is a separate key management service that is air gapped. This is used to generate the keys and certificates necessary for secure communication and to ensure integrity during exchange of data.

There is also a logging infrastructure that collects data from each subsystem. Each subsystem generates their own immutable logs locally and the central system receives copies of the logs continuously. In addition, the central system receives messages from the infrastructure for monitoring purposes.

To support counting of paper ballots, there is a separate system installed in municipal counting centres (est. 150 centres). This is the eCounting system. It contains modules for scanning and verifying ballots, in addition to local administrative functions.

Systems use web or web services to exchange data through secure channels.

3. Interfaces between main system components

3.1 Distribution of Election configuration

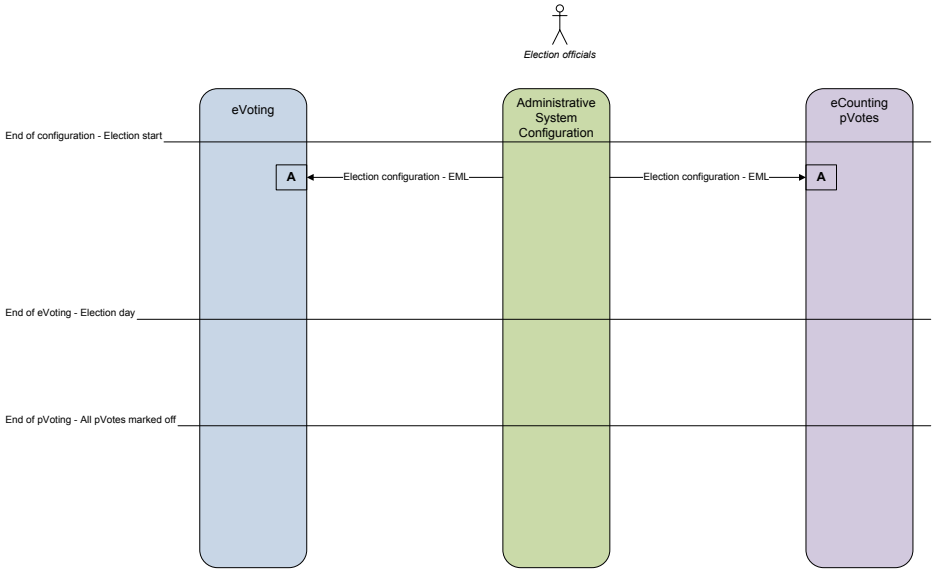


Figure 1: DFD Admin Configuration context level view

The election configuration is distributed to the other subsystems through downloads of EML files.

3.2 List proposals

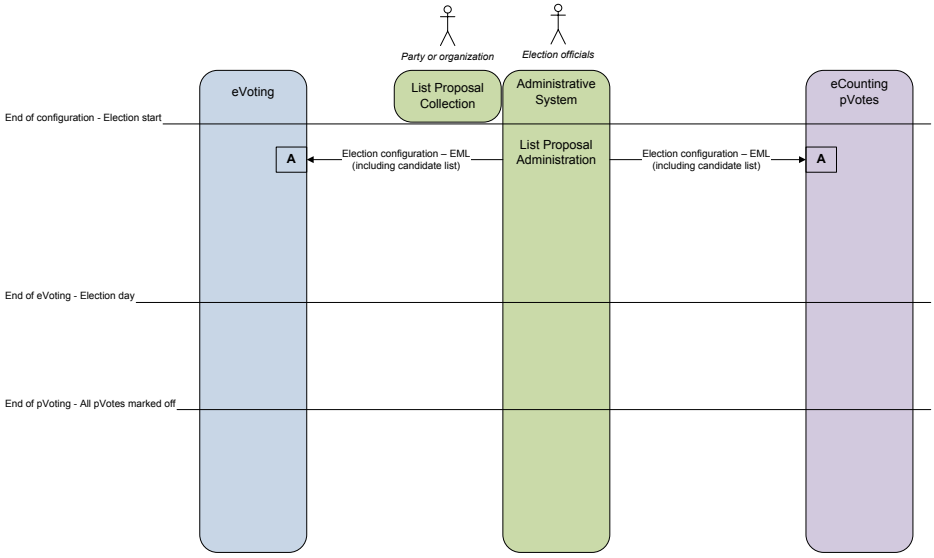


Figure 2: DFD List Proposal context level view

Candidates are added to the election through a party list system where parties submit lists for approval. When approved, candidate lists are distributed to the other subsystem as part of the EML distribution.

3.3 Distribution of Electoral roll

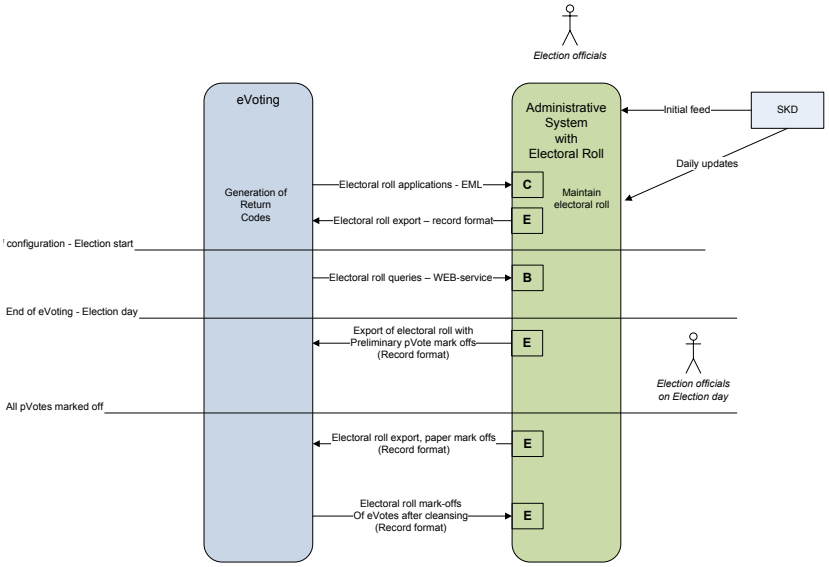


Figure 3: DFD Electoral Roll context level view

The electoral roll is only distributed to the electronic voting systems. Paper based voting interface directly with the administration system through the poll book application.

Voters can request to be added to the electoral roll through the eVote system. Applications are transferred to the administration system as EML files.

Prior to the election, the eVote system receives a copy of the electoral roll to generate return codes. During the election the eVote system will query the administration system to verify that a voter is in the electoral roll.

At the end of eVoting a new export the electoral roll is distributed to allow the eVote system to again verify all votes against the electoral roll and to eliminate any voters that have voted on paper. This is done in several iterations to allow for preliminary result reporting.

3.4 Upload of counts

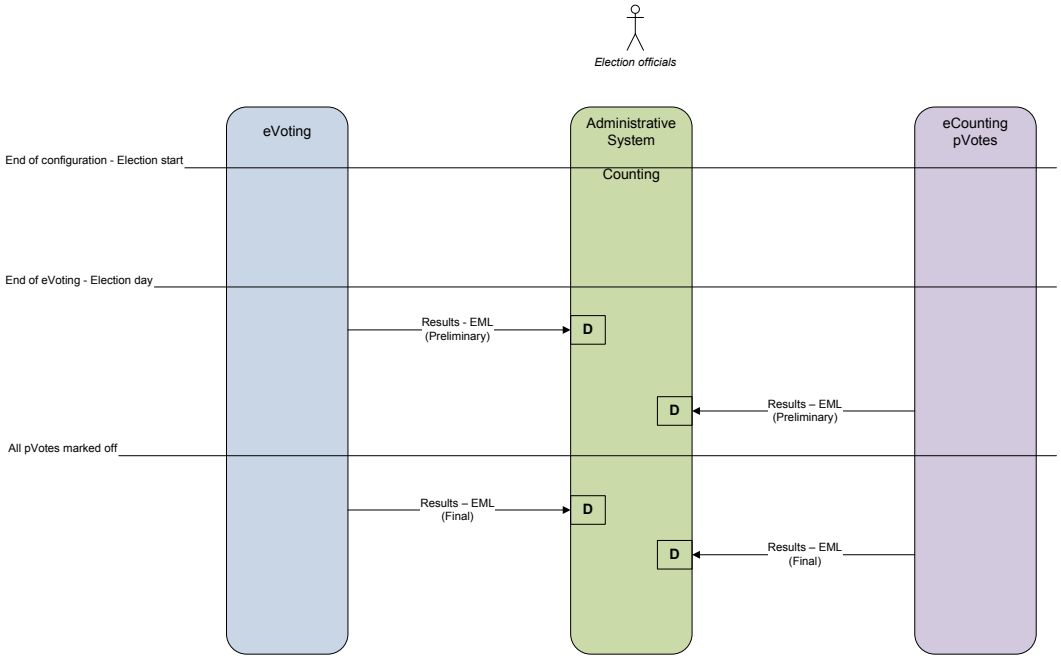


Figure 4: DFD Admin Counting context level view

Counts are uploaded to the administration system as signed EML files. Results can be uploaded several times. The final counts are established when all paper votes have been counted.

4. File transfer and formats between main system components

4.1 General on Signing

All files exchanged between the domains must be signed. It's the applications that create the result file that is responsible for creating the signature. All applications that receive files are responsible for verifying the signature.

4.2 Message Formats

All the data exchanged between the domains are done by using the Election Markup Language (EML).

The following table shows the message formats in use and witch domains they are exchanged between:

Format	Channel A	Channel B	Channel C	Channel D	Channel E
EML 110	X		X		
EML 230	X		X		
EML 460				X	
EML 510		X		X	
Record format (for electoral roll export)					X

110 – Election Event
This message provides information about the election event, all elections within that event, and all contests for each election. Please refer Appendix A for an example.

230 – Candidate List
For each contest, this message contains a list of candidates or a list of parties, optionally with a list of candidates under each party. Each candidate or party in the Candidate List corresponds to a ballot in the contest. Please refer Appendix A for an example.

460 – Votes
This message shall be used to transfer information about individual votes. In our context these votes must be anonymized. This message type is always used for transferring proposed rejected ballots from eCounting of pVotes. The message shall also be used for transferring information on each corrected ballot. This applies to both eVote and pVote counting results.

Please note! A Votes EML-message is uniquely connected/related to a corresponding Count message. Thus, the transaction ID in the two EML-messages must be the same for a specific counting result.

510 – Count
This message is used to transfer the counting results at the relevant counting level. One file can only contain results from one counting level (e.g. only results from one voting district, if not counting results are aggregated at municipality level). The file contains the number of each type of ballot. Please note that for votes that are counted at the district level, the results from all polling districts must be reported, even though no votes have been casted in the district.

Refer Redmine issue https://nwdevel.scytl.net/redmine/issues/1300 for details.

Electoral roll - record format
The eVote system operators shall be able to download a file that holds the mark offs of submitted pVotes in the electoral roll. This file will be input to the cleansing process when counting eVotes. The same format is used when the eVote platform uploads the eVote mark-offs.

In this file there are records with the following format (comma separated):

- ElectionEventIdentifier
- ElectionGroupIdetifier
- ElectionIdentifier
- ContestIdentifier
- VoterIdentification
- TypeOfMarkOff (valid values: pvoting, evoting, evotingctrl)
- VoteCasted
- Area Idetifier (which holds the counting area identifier at the lowest level)

For example:
```
200701,1,2,101,1010731867,pvoting,TRUE,47.01.0101.010100.0001
200701,2,4,47,1010731867,pvoting,FALSE,47.01.0101.010100.0001
```

There will be one record for each voter in the electoral roll for each of the elections in the current election event. (Example: If there are three different elections within and election event, there will be 3 records for each voter showing the mark-off for each election).

Refer Redmine issue https://nwdevel.scytl.net/redmine/issues/4147 for details.

4.3 File transfer of election configuration

The configuration files are signed individually by the Admin system. All files are zip'ed into one "transfer-file" that may be downloaded. It is up to the receiving system to verify the signatures on the configuration file(s) they are using.

The structure of the zip file is:

- Zip file (i.e. configuration.zip)
  - ElectionEvent.xml
  - ElectionEvent.pem (signature)
  - CandidateList.xml
  - CandidateList.pem (signature)
  - Municipalities.csv
  - Municipalities.pem (signature)

Please note that there will be only be configuration data related to **one** election event in the zip-file.

4.4 File transfer of counting results

The counting results are group into one zip file and the signature is stored in another external file. To assure that the result zip-file and the signature file is kept together in the transfer between the different systems, these two files are zipped again into one transfer file.

As described above the message type "510–Count" and the corresponding message type "460-Votes" shall always be transferred in the same Zip file and have the same transaction ID in the messages.

The structure of the zip file is:

- Zip file (i.e. Transfer.zip)

- ZIP file containing counting results, the file name should be unique and the extension MUST be .zip (i.e. Counts.zip)
  - Count-0001.xml
  - ....
  - ….
  - Count-nnnn.xml
  - Votes-0001.xml
  - ….
  - ….
  - Votes-nnnn.xml
  - Optional directory if images that are to be transferred .\images
    - BallotID.tiff
    - …..
    - BallotID.tiff
- Counts.zip.pem (signature file)

As indicated above, several counting results may be transferred in the same zip file. But note the restriction that requires a pair of Counts/Votes files to have the same, and unique, transaction ID.

4.5 Transfer of electoral roll, optionally with mark-offs, from Admin to eVote

As described above a record format is used to transfer the electoral roll with or without mark offs. There will be one file for each municipality. A digital signature will be applied to each of the files. All files are group together in one zip file for transfer purposes.

The structure of the zip file is:

- Zip file (i.e. configuration.zip)
  - ElectionList_0101.xml (electoral roll with mark offs for municipality 0101)
  - ElectionList_0101.pem (signature)
  - ElectionList_0301.xml (electoral roll with mark offs for municipality 0301)
  - ElectionList_0301.pem (signature)
  - ElectionList_0906.xml (electoral roll with mark offs for municipality 0906)
  - ElectionList_0906.pem (signature)
  - …….

Please note that for the preliminary count of eVotes, all municipalities will be present. For the final count of eVotes, an export will be generated at regular intervals (e.g. every second hour). In this case only part of the ER is exported; for the municipalities that have signaled that all paper votes have been cast. The eVoting mark offs transferred at the end of the process has all municipalities present (where eVotes have been cast).

4.6 Usage of virtual voting districts to preserve anonymity

To preserve anonymity of the voter there are two mechanisms that are used:

1. According to the Norwegian election law voting districts with less than 100 voters cannot be counted separately. When the municipality configures the election, they have a feature in the Admin system that allows them to create a special virtual counting district and connect small "physical" voting districts to that one. This virtual counting district is thus predefined and its name and ID is transferred along with information on all other polling districts in the candidate EML-file. In the candidate EML it is shown that the physical (small) polling districts are associated with the virtual one:

```
<PollingDistrict Id="47.02.0215.021500.0004">
        <Name>Dal</Name>
        <Association Id="47.02.0215.021500.8999">Samlekrets valgstyret</Association>
</PollingDistrict>
<PollingDistrict Id="47.02.0215.021500.0005">
        <Name>Digerud</Name>
<Association Id="47.02.0215.021500.8999"> Samlekrets valgstyret </Association>
</PollingDistrict>
<PollingDistrict Id="47.02.0215.021500.8999">
        <Name>Samlekrets valgstyret</Name>
</PollingDistrict>
```

   As seen from the above example the physical polling districts "0004" and "0005" is associated with the virtual polling district "8999", and thus should not be counted separately but combined in polling district "8999". This situation, i.e. when the virtual counting district has been defined in advance is determined by the fact that the polling districts are associated with a virtual district at the same level in the geographical hierarchy.

2. A special mechanism shall be implemented in the eVoting counting process, which assures that all counting reported shall have a minimum number of votes. Initially this number is set to 20, but the number must be possible to set as a system parameter in a configuration of the eVote counting platform. If the eVoting counting process finds a polling district with less than 20 votes, it shall not count the votes in that polling district, but count the votes in a virtual polling district at the municipality level (i.e. actually the "Bydel" level; the level above the polling district itself). Obviously to assure that the virtual polling district has at least 20 votes, the counts from some other physical polling districts must be added to the virtual one as well. All municipalities (actually "Bydel") have always a virtual polling district. This polling district is marked in the following way in the candidates EML:

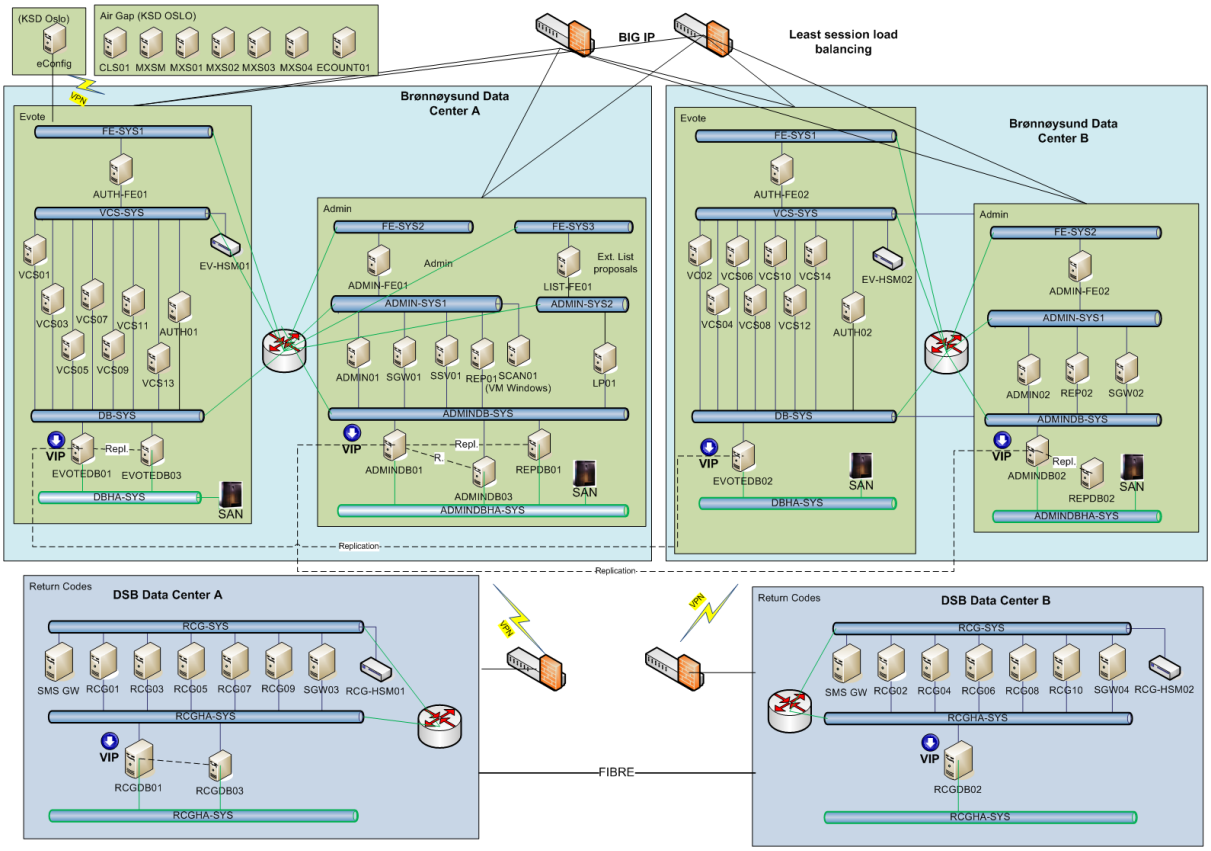
```
<PollingDistrict Id="47.02.0216.021600.0000">
    <Name>Uoppgitt</Name>
    <Association Id="47.02.0216.021600">Uoppgitt</Association>
</PollingDistrict>
```

   Please note that the association in this case point one level up (i.e. the "Bydel" 021600).

When method 2 above is used, the physical polling districts where no counting is performed should always be reported in the count EML. Refer the "Count-empty.xml" on Redmine issue #1300 to see how a polling district that rolls up its votes is signaled.

5. Deployment

5.1 Infrastructure overview eVoting and Administration system



The E-vote 2011 infrastructure is divided between four data centers. The main data centers are located in Brønnøysund and configured in a load balanced / fail-over setting through a BIG-IP cluster. The data centers for generation of return codes are physically separated and located at DSB. The functional areas of the system are divided into separate network zones for added security. The systems use a three layer architecture where front-end severs are separated from the application and database servers.

5.1.1 Oslo (KSE)

Subsystem	Server name	Description	Server type	Base config
Evote	CNT01	Support server for extracting data for air gapped counting	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Evote Config air gapped	eCONFIG Laptop01,02,03,04	Election configuration Laptops	Intel (non-Atom!) 4GBRAM TPM 250GB	CentOS
Evote Config air gapped	eCONFIGSPARE Laptop01	Election configuration Spare Laptop	Intel (non-Atom!) 4GBRAM TPM 250GB	CentOS
Evote air	eCL01,02	Cleansing server to	Dell R710, 16 GB	CentOS

gapped		eliminate votes that are not to be counted	RAM, 2x500GB SAS 15k disks **8 Cores**	Glassfish
Evote air gapped	eMIX01,02,03,04	Mixing servers to shuffle votes.	Dell R710, 16 GB RAM, 2x500GB SAS 15k disks **8 Cores**	CentOS Glassfish
Evote air gapped	eCOUNT01,02	eCounting	Dell R710, 16 GB RAM, 2x500GB SAS 15k disks **8 Cores**	CentOS Glassfish
Evote air gapped	eMIXSPARE01,02	Spares for Cleansing/Mixing/Counting	Dell R710, 16 GB RAM, 2x500GB SAS 15k disks **8 Cores**	CentOS Glassfish
Evote air gapped	eMIXLAPTOP01,02	Laptops to operate Cleansing/Mixing/Counting	Intel (non-Atom!) 4GBRAM TPM 250GB	CentOS
Evote air gapped	EVMIX-HSM01,02	Hardware Security Device	N/A	N/A

5.1.2 Data Center 1 (Brønnøysund)

Subsystem	**Server name**	**Description**	**Server type**	**Base config**
Evote	AUTH-FE01	Front end for electronic voting. Redirects authentication requests to DIFI (Min ID)	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Evote	VCS01, 03…, 13 (7 servers)	Vote collector servers. Receives and encrypts votes	Dell R810, 32GB RAM, 2x146GB SAS 15k disks **12 cores**	CentOS Glassfish
Evote	AUTH01	Back-end for authentication (Min ID)	Dell R810, 32GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Evote	EVOTEDB01	Part of cluster EVOTEDB01 and EVOTEDB02 (in the other data center). Data is replicated.	Dell R710, 16GB RAM, 2x146GB SAS 15k disks SAN	CentOS Postgres
Evote	EVOTEDB03	Cold standby in case cluster fails. Data is replicated form EVOTEDB01	Dell R710, 16GB RAM, 2x146GB SAS 15k disks SAN	CentOS Postgres
Evote	EV-HSM01	Hardware Security Device	N/A	N/A
Admin	ADMIN-FE01	Front end for administration system	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Admin	LIST-FE01	Front end for external list proposals (used by parties to submit	Dell R610, 8GB RAM, 2x146GB SAS 15k	CentOS Glassfish

		candidate lists)	disks	
Admin	ADMIN01	Application server for administration system	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Admin	SCAN01	ICR interpretation of list proposals (signatures)	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	Windows ReadSoft
Admin	LP01	List proposals back-end	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Admin	ADMINDB01	Database for administration system including Electoral Roll. In cluster configuration with ADMINDB02 in other datacenter. Data is replicated from 01 to 02	Dell R710, 16GB RAM, 2x146GB SAS 15k disks SAN	CentOS Postgres
Admin	ADMINDB03	Cold Standby, data is replicated from 01	Dell R710, 16GB RAM, 2x146GB SAS 15k disks SAN	CentOS Postgres
Auditing	AUD01	Auditing Immutabilizator	Dell R710, 8 GB RAM, 2x1TB SAS 15k disks **8 Cores**	CentOS Glassfish

5.1.3 Data Center 2 (Brønnøysund)

Subsystem	**Server name**	**Description**	**Server type**	**Base config**
Evote	AUTH-FE02	Front end for electronic voting. Redirects authentication requests to DIFI (ID-porten)	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Evote	VCS02, 04…, 14 (7 servers)	Vote collector servers. Receives and encrypts votes	Dell R810, 32GB RAM, 2x146GB SAS 15k disks **12 cores**	CentOS Glassfish
Evote	AUTH02	Back-end for authentication (ID-porten)	Dell R810, 32GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Evote	EVOTEDB02	Part of cluster EVOTEDB01 and EVOTEDB02 (in the other data center). Data is replicated.	Dell R710, 16GB RAM, 2x146GB SAS 15k disks SAN	CentOS Postgres
Evote	EV-HSM02	Hardware Security Device	N/A	N/A
Admin	ADMIN-FE02	Front end for administration system	Dell R610, 8GB RAM, 2x146GB SAS 15k	CentOS Glassfish

			disks	
Admin	ADMIN02	Application server for administration system	Dell R610, 8GB RAM, 2x146GB SAS 15k disks	CentOS Glassfish
Admin	ADMINDB02	Database for administration system including Electoral Roll. In cluster configuration with ADMINDB02 in other datacenter. Data is replicated from 01 to 02	Dell R710, 16GB RAM, 2x146GB SAS 15k disks SAN	CentOS Postgres
Admin/ Reporting	REP02	Reporting WS	Dell R710, 8 GB RAM, 2x146GB SAS 15k disks **8 Cores**	CentOS Glassfish
Admin/ Reporting	REPDB02	Reporting Database	Dell R710, 8 GB RAM, 2x146GB SAS 15k disks **8 Cores**	CentOS Postgres
Auditing	AUD02	Auditing Immutabilizator	Dell R710, 8 GB RAM, 2x1TB SAS 15k disks **8 Cores**	CentOS Glassfish

5.1.4 Data Center 3 (DSB)

Subsystem	**Server name**	**Description**	**Server type**	**Base config**
Evote	SMS-GW	SMS Gateway to transmit return codes to voter	SysManSMS	N/A
Evote	RCG01, 03…, 09 (5 servers)	Return code generators	HP xxx, 32GB RAM, 2x146GB SAS 15k disks TPM **8 Cores**	CentOS Glassfish
Evote	RCG-HSM01	Hardware Security Device	N/A	N/A
Evote	RCGDB01	Database for return code system In cluster configuration with RCGDB02 in other datacenter. Data is replicated from 01 to 02	HP xxx, 16GB RAM, 2x146GB SAS 15k disks TPM SAN	CentOS Postgres
Evote	RCGDB03	Cold Standby, data is replicated from 01	HP xxx, 16GB RAM, 2x146GB SAS 15k disks TPM SAN	CentOS Postgres
Auditing	AUD03	Auditing Immutabilizator	Dell R710, 8 GB RAM, 2x1TB SAS 15k disks **8 Cores**	CentOS Glassfish

5.1.5 Data Center 4 (DSB)

Subsystem	Server name	Description	Server type	Base config
Evote	SMS-GW	SMS Gateway to transmit return codes to voter	SysManSMS	N/A
Evote	RCG02, 04…, 10 (5 servers)	Return code generators	HP xxx, 32GB RAM, 2x146GB SAS 15k disks TPM **8 Cores**	CentOS Glassfish
Evote	RCG-HSM02	Hardware Security Device	N/A	N/A
Evote	RCGDB02	Database for return code system In cluster configuration with RCGDB02 in other datacenter. Data is replicated from 01 to 02	HP xxx, 16GB RAM, 2x146GB SAS 15k disks TPM SAN	CentOS Postgres
Auditing	AUD04	Auditing Immutabilizator	Dell R710, 8 GB RAM, 2x1TB SAS 15k disks **8 Cores**	CentOS Glassfish

5.1.6 Base Software configuration

The following software configuration is used as base install. Please refer to the table above for details of what base configuration is located in the servers.

Base component	Product	Current version as of 07.12.2010
Operating System	CentOS	5.5
Application server	Glassfish Open Source Edition	3.0.1
Windows server	Windows	2008
Database	Postgres	9.0
ICR scanning	ReadSoft	5.2

5.1.7 Network and infrastructure security

Data centers are secured with redundant firewalls. In addition, servers are hardened to accept connections only towards servers shown in the connectivity diagram. In addition, virtual LANs are used to filter communication within the LAN zones. Servers are stripped for any services that do not relate to the operation of the system.

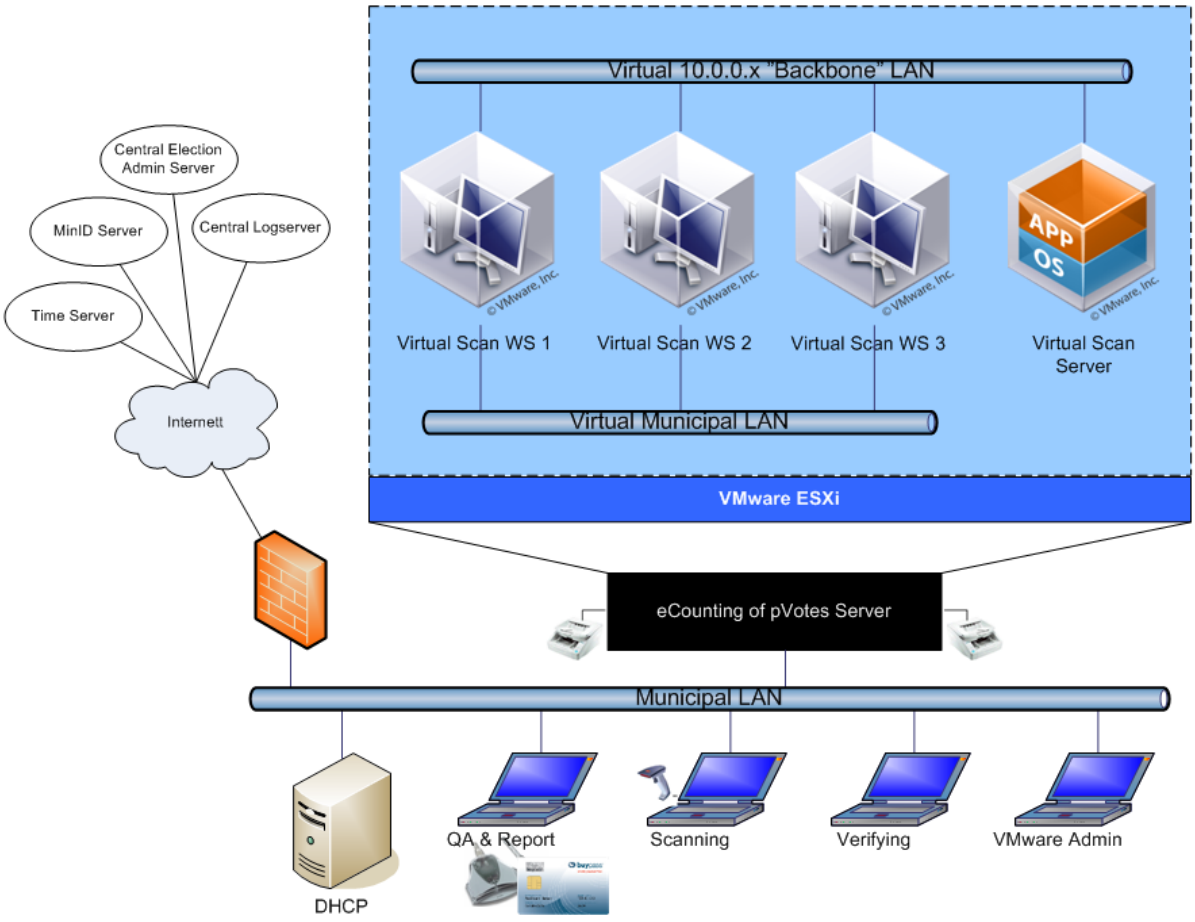
Communication between data centers 1 & 2 are redundant fixed lines. Communication between data centers 3 & 4 are redundant fixed lines. Data centers 1 & 2 communicate with data centers 3 & 4 through redundant VPN lines.

Communication to external entities is secured as follows:

- SKD, for electoral roll updates, use fixed line
- SSB, for results reporting, use web services over internet with HTTPS and keys
- Counting centers use web services over internet with HTTPS and keys

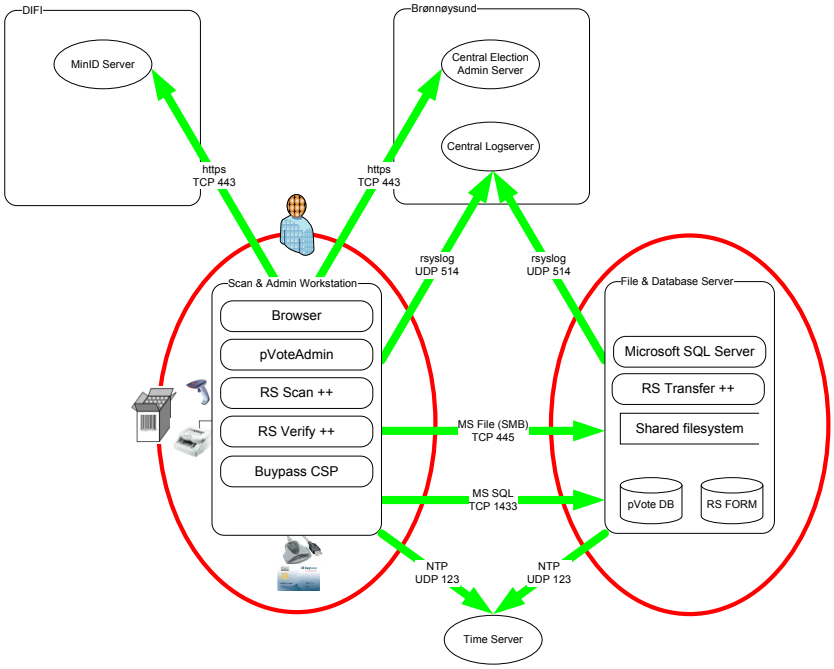
5.2 Infrastructure overview eCounting system

The workstation, servers, and infrastructure for eCounting of pVotes will be deployed as virtual machines in a VMware ESX system:



The virtual system is documented in a separate document. Below is the scanning workstation and servers described as standalone machines.

5.2.1 Configuration



The eCounting infrastructure is located in the counting centers in the municipalities. The basic configuration has two computers: Scan & Admin workstation and File & Database server. The configuration can be increased with more Scan & Admin workstations as necessary.

The scan & admin workstation is connected to a ballot scanner and a bar code reader. In addition it is connected to a smartcard solution to sign counts.

As standalone solutions, the workstations must have the following minimum configuration:

Workstation	Minimum hardware configuration
Scan & Admin workstation	Pentium III, 2GB memory, 250GB hard drive
File & database server	Pentium III, 2GB memory, 250GB hard drive

The PCs must have network access.

5.2.2 Base Software configuration

The following software configuration is used as base install. Please refer to the table above for details of what base configuration is located in the servers.

Base component	**Product**	**Current version**
Operating System	Windows 2008 server, Windows 2008 sufficient on Scan & Admin workstation	2008
Application	ReadSoft	5.2
Database	SQL Server 2008	2008
Workstation	Windows 7, 32-bit	7

5.2.3 Network and infrastructure security

Each counting center is secured by the local municipality system administrator. The administrator is responsible for securing the PC to ensure that the Windows software is patched and virus free. In addition, the administrator is responsible for limiting access in the municipality firewall.

In addition, each PC will be secured in the local firewalls. Only ports shown in the diagram will be permitted for use in the direction indicated.

5.3 Securing connectivity between systems

5.3.1 Data exchange points

Information flows between the following components:

- Municipal networks – Admin System.
- Election sites – Admin system.
- eVoting components – Admin system.
- pVoting components – Admin system.
- eVoting components – Electoral Roll.
- Admin system. – SSB.
- Admin system. – SKD.

The following communications channels need secure communications in this context:

1) Election administrators' use of the Admin system from municipal networks prior to elections.
2) Election administrators' use of the Admin system from election sites during the election
3) Download of election configuration to eVote system
4) Download of election configuration to pVote systems
5) Queries into Electoral Roll (ER) from the eVote system during elections.
6) Download of ER mark-offs to eVote system
7) Submission of counts (results) from the pVote systems during elections
8) Submission of counts (results) from the eVote systems during elections
9) Transfer of results to SSB during and after elections
10) Transfer of Electoral Roll data from SKD
11) Public access to list proposals

5.3.2 Mutual authentication of systems

Therefore, the following items must are considered as system components that need the use of mutually authenticated communications (SSL certificates are needed in these machines):

- Machines from Municipal networks.
- Machines from Election sites.
- eVoting modules in charge of accessing to the Admin System or the Electoral Roll:
    - eVoting Administration module.
    - eVoting Authentication Service (uses Electoral Roll).
    - eVoting Counting module.
- eCounting modules in charge of accessing to the Admin System:
- SSB.
- SKD

5.3.3 Securing sessions

HTTPS requires distribution of valid certificates. A secure distribution is of importance.

With MinID-authentication the user identity is fairly certain, particularly with level 4. The need to authenticate the client arises with the client PC. A client PC can be infected with Trojans or key loggers that misuse the secure sessions.

It is therefore important that the client PCs are securely managed by the municipality administrators and that

sufficient anti-virus scanning and management of policies is performed.

5.3.4 Public access

Applications that provide public access cannot be secured as described above. Therefore isolation of resources is the only means and one cannot combine internal web applications and external web applications in the same application framework.

5.3.5 Certificate management

Between system components residing in data center environments, certificates are distributed as part of the installation process. For system components that are residing outside the data centers certificates are generated by responsible system administrators and distributed by removable media via mail.

Certificates will be managed by region, where a region is represented by a system administrator in the municipality. This means each municipality will be issued one certificate. It is the local system administrator's responsibility to personally install this certificate on selected PCs, and secure the PC such that the certificate cannot be exported.

If the certificate should become compromised, the local system administrator is responsible for requesting a new certificate to be issued. Key and certificate management are further covered in the eVoting architecture document and the Security architecture documents.

5.3.6 Methods for securing clients

There are different certificates between each category, but shared within each category:

Communication requirement	**Connection**	**Solution**	**Comment**
Election administrators' use of the Admin system from municipal networks prior to elections.	Admin web interface	HTTPS with certificate	Municipalities responsible for client security Shared certificate
Election administrators' use of the Admin system from election sites during the election	Admin web interface	HTTPS with certificate	Municipalities responsible for client security Shared certificate
Download of election configuration to eVote system	Admin web interface download	HTTPS with certificate + switch	Internally in data center
Download of election configuration to pVote systems	Admin web interface download	HTTPS with certificate	Shared certificate installed as part of virtual image (if change order approved)
Queries into Electoral Roll (ER) from eVote system during elections.	Web service	HTTPS with certificate + switch	Internally in data center
Download of ER mark-offs to eVote system. Note that mark-offs are manually moved to air gapped systems from the eVote system.	Web service	HTTPS with certificate + switch	Internally in data center
Submission of counts (results) from the pVote systems during elections	Admin web interface upload	HTTPS with certificate	Shared certificate installed as part of virtual image
Submission of counts (results) from the eVote systems during elections	Admin web interface upload	HTTPS with certificate + switch	Internally in data center
Transfer of results to SSB during and after elections	Web service	HTTPS with certificate	Data center to data center
Transfer of Electoral Roll data from SKD	SFTP and fixed line	HTTPS with certificate	From external data center
Public access to list proposals	Separate server	HTTPS (with server certificate)	Only user authentication

Appendix A: EML-config files

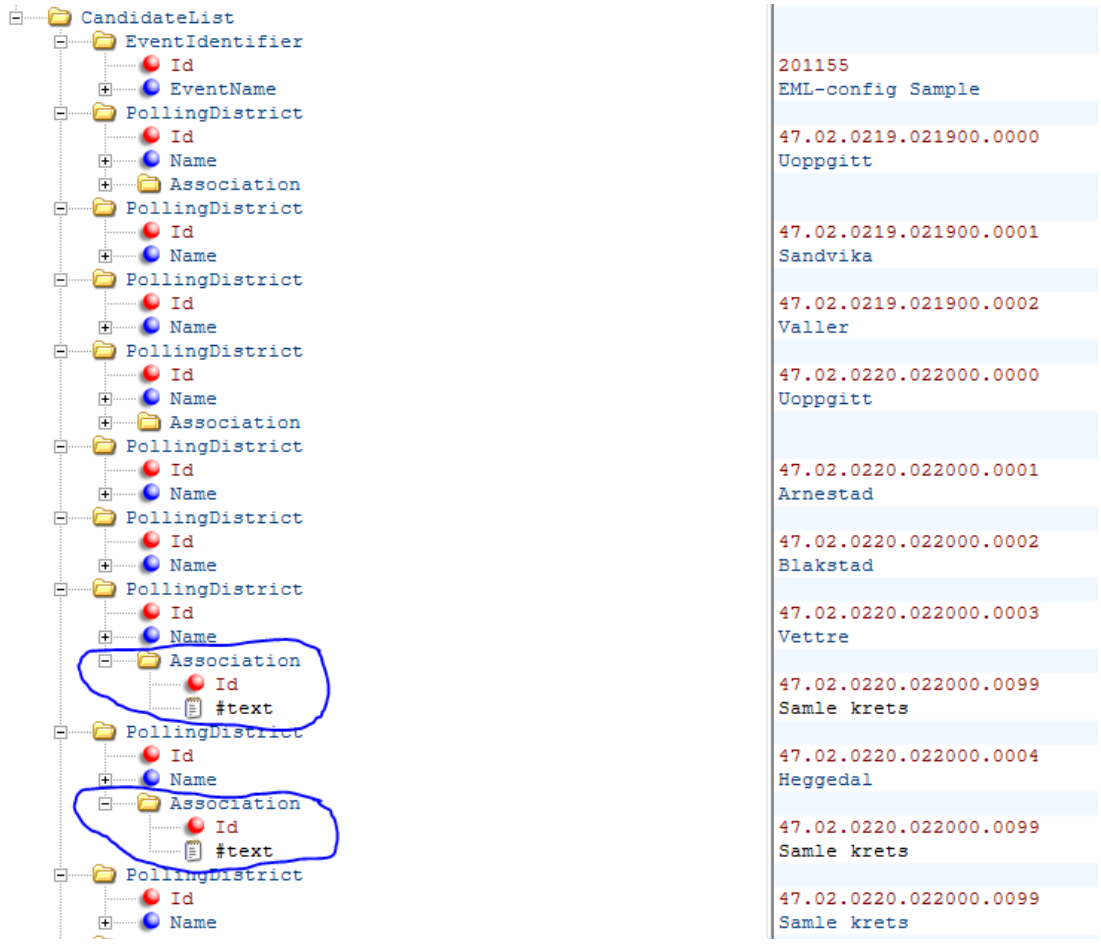
This Appendix gives an example of a typical configuration of a County- and Municipality election. We have made a simple configuration to illustrate the concept of the two configuration files:

- ElectionEvent.xml
- CandidateList.xml

Geographical configuration of the sample ElectionEvent:

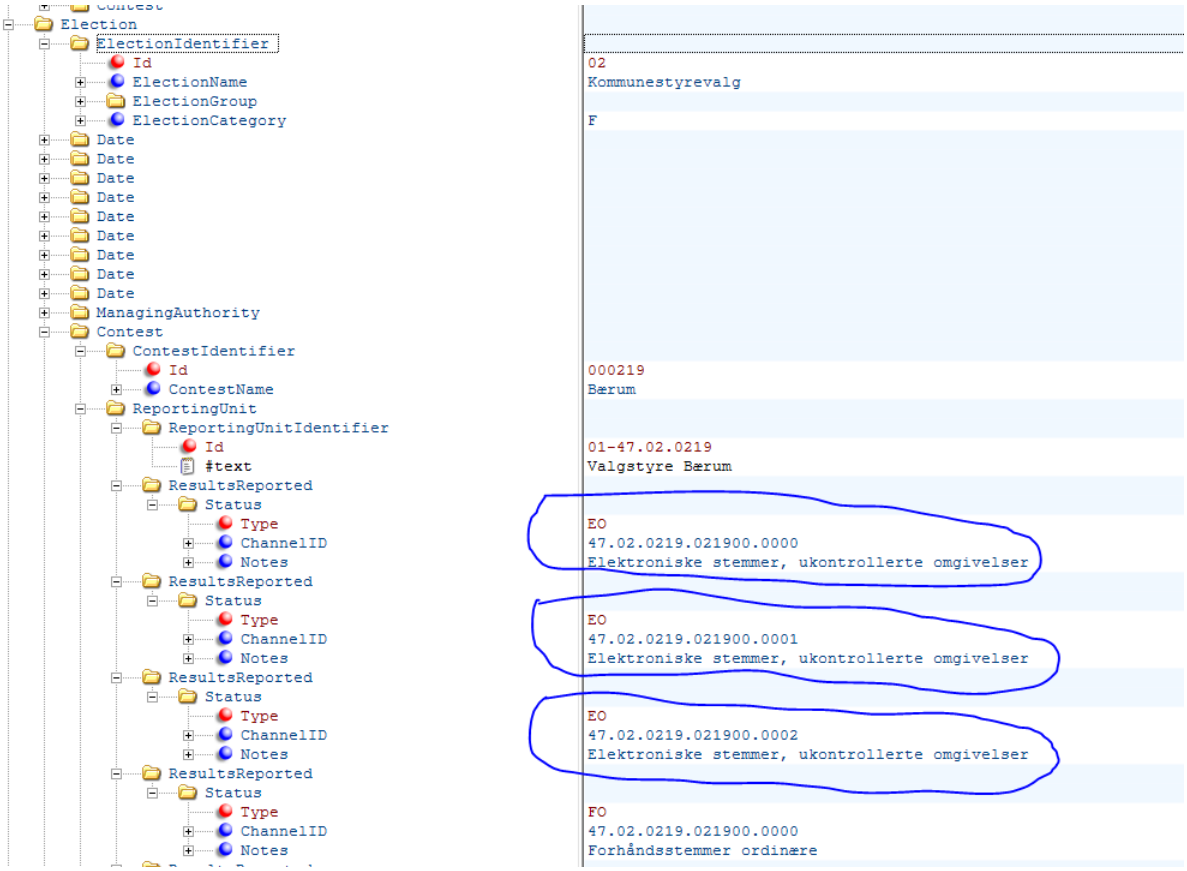
Country	**County**	**Municipality**	**Borrow**	**Polling district**
47 – Norway	02 – Akershus,	0219 –Bærum,	021900 – Ingen bydel	0000 – Uoppgitt 0001 – Sandvika 0002 - Valler
		0220 – Asker	022000 – Ingen bydel	0000 – Uoppgitt 0001 – Arnestad 0002 – Blakstad 0003 – Vettre 0004 – Heggedal 0099 – Samle krets

The Polling district “0099-Samle krets” in Asker is a predefined virtual polling district where the votes from “0003 – Vettre” and “0004-Heggedal” shall be counted together. This information is found in the CandidateList.eml:



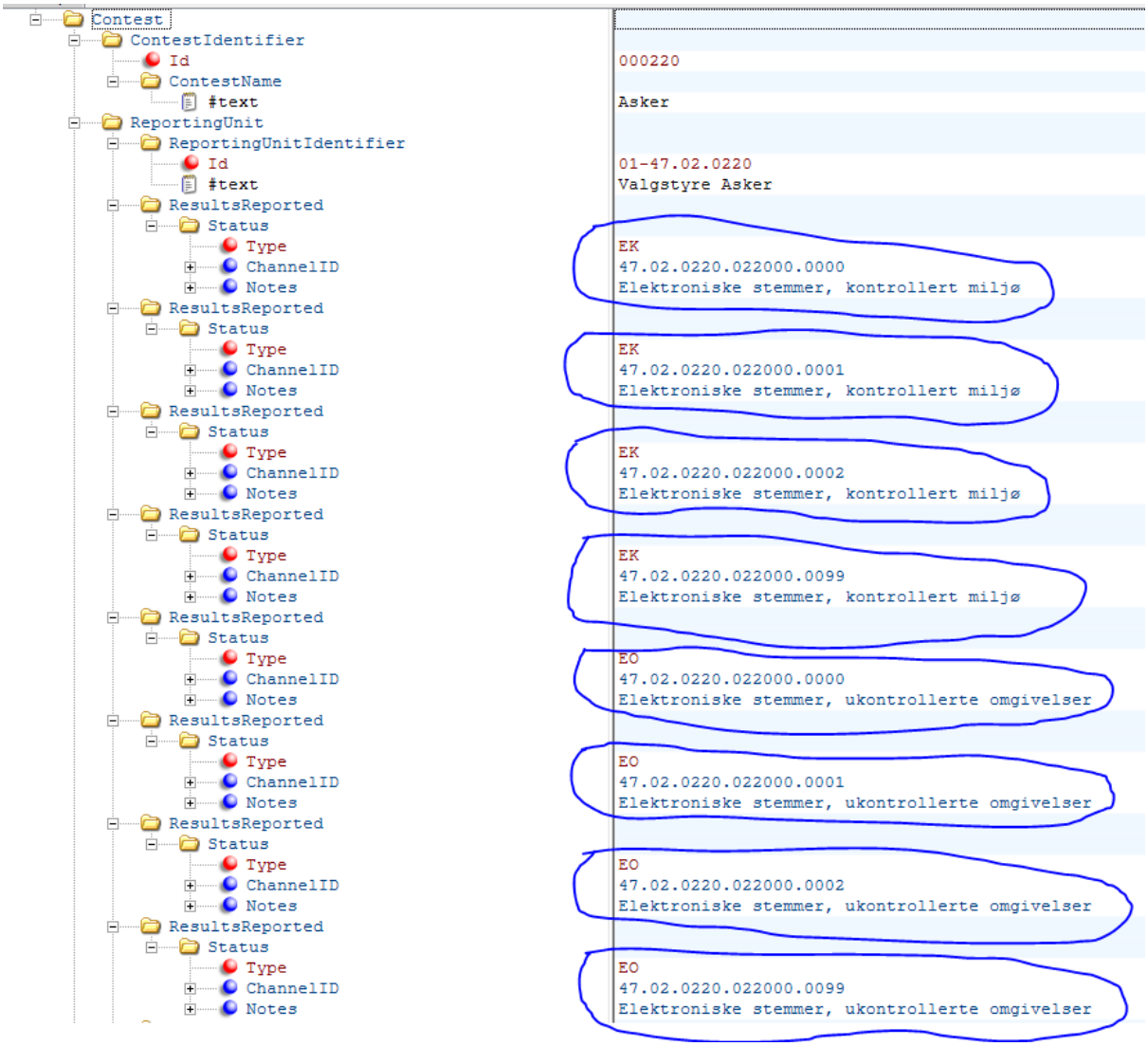
The Bærum municipality is configured for ONLY eVote in uncontrolled environment, but the Asker municipality is configured for eVoting in both controlled and uncontrolled environment. Which type of votes that should be reported when counting, and for which polling districts is found in the ElectionEvent.eml:

As described above, Bærum is configured for eVote in uncontrolled environment only in addition to the different pVote Categories:



Thus, in this (simple) case, the eVote platform shall report results only for category “EO” and for the following polling districts: 0000, 0001 and 0002.

As described above, Asker is configured for both eVote in uncontrolled- and controlled environment (in addition to the different pVote Categories). In Asker a virtual counting district is also defined holding the counts for the physical polling district 0002 and 0004. The drawing on the next page shows a similar EML extract that shows which type of counting category that shall be reported for Asker and for which polling districts.



As shown for Asker both “EK” and “EO” shall be reported for polling districts: 0000, 0001, 0002 and 0099 (consisting of 0003 an 0004).