E-vote 2011

SSA-U Appendix 2A

Contractor Solution Specification

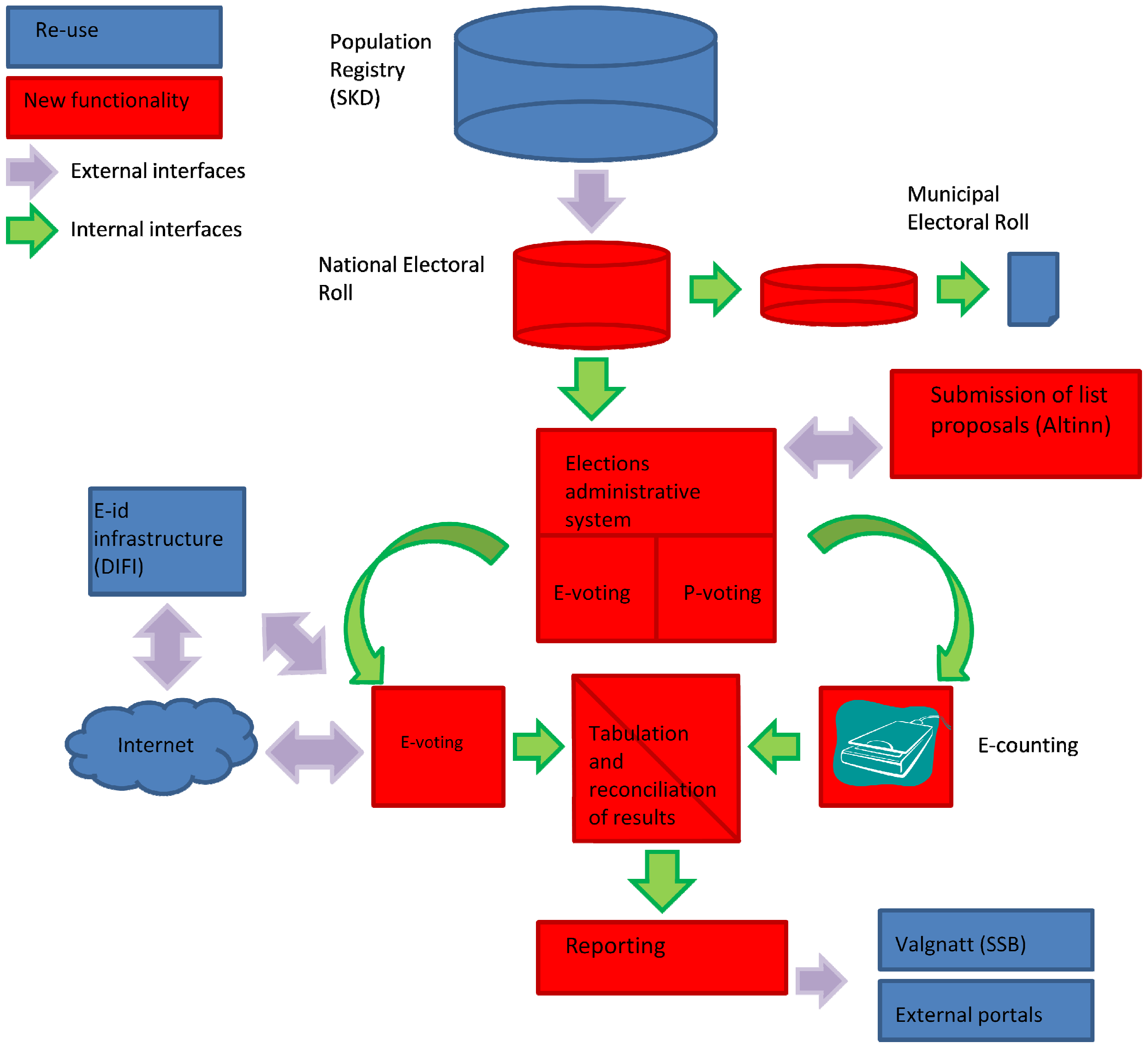
Project: E-vote 2011

CONTENT

1. Tenderer Solution Proposal

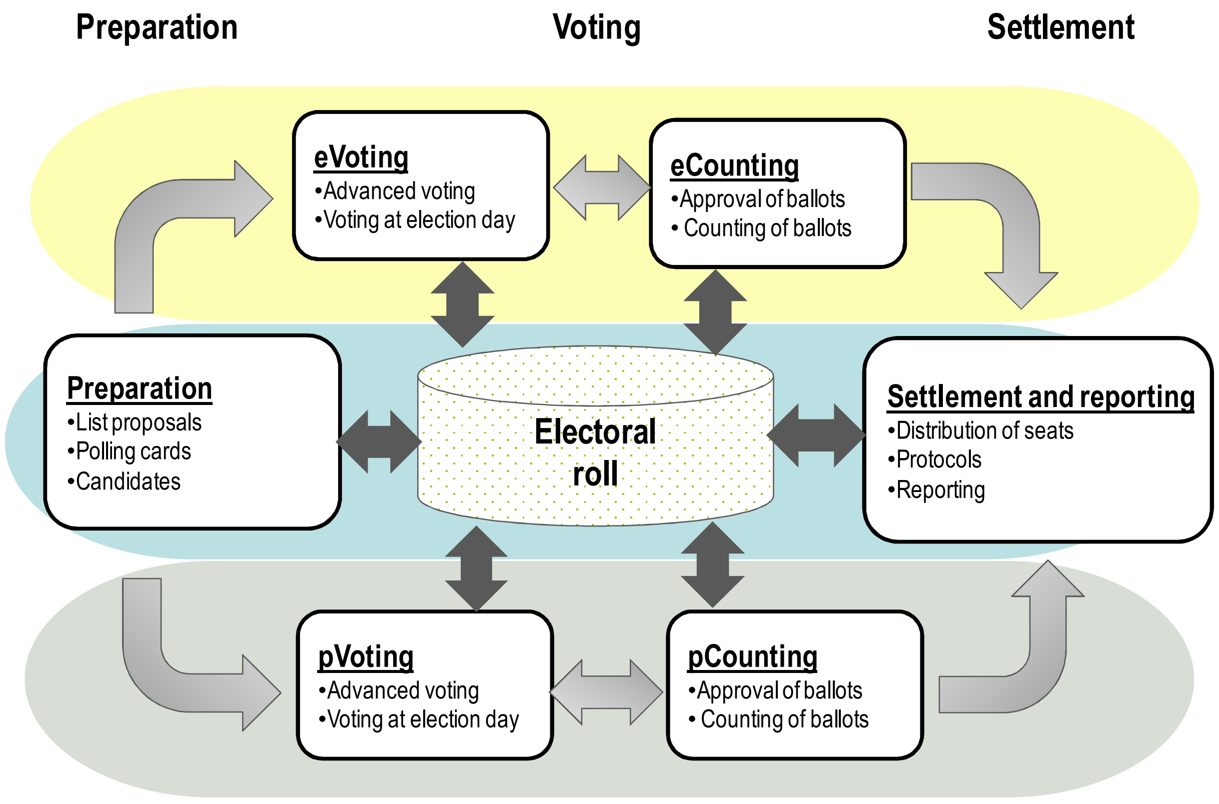
1.1. Overview of Proposed Functional Solution

This section is intended to provide the Ministry with an overview of the complete voting solution proposed by The Contractor. In addition to this overview each component is described in further detail in its corresponding Use Case. The drawing below shows the conceptual model provided by KRD. This drawing is added in this functional description mainly for reference purposes and for the purpose of showing related external systems that are to be interfaced with the proposed solution.



Functional model overview

The drawing below shows the overall functional model that reflects the different phases that cover the complete election process.



The Preparation phase

The main purpose of this phase is to support all preparations that are needed. This include preparations that are needed before the voting can start, as well as preparations necessary for the counting- and election settlement process.

The election administrative system provides the management of different kinds of elections such as municipality, county, Sami and general elections as well as referendum, citizen consultation, and so forth.

Managing the electoral roll

The basis for the electoral roll is received as an extract from the national population register. This roll is also continuously updated by changes received from the national population register within the administrative timeframe of the election. The electoral roll forms up the heart of the system where all authorized voters are registered. The system supports the following tasks:

- Add voters to the electoral roll based on well-defined procedures for citizens that apply for voting.
- Change characteristics of voters if necessary.
- Delete voters from the electoral roll.
- The system keeps a log that shows all changes that have been applied to the electoral roll. All changes done to a specific voter or more general statistics at the municipality or district level are displayed on request.
- The system has functions to mark that a polling card has been returned. This is likely to happen if the voter's address in the electoral roll is not correct.
- Printouts of electoral roll.

Voting districts and voting stations

The use of voting districts is configurable. Today, elections include voting districts. Without this structure there may be some implications:

- It will not be possible to provide the statistics that are provided to day on the voting district level. (The statistics will not be accurate).
- The fall-back procedure on each polling station, when connection with the central electoral roll is lost, will be less automated

Party registration and nomination process

The system fully handles the process of receiving and controlling party lists and candidates. This is a strict administrative process which involves control towards the electoral roll as well as informing the candidates that they are nominated. Examples of functions supported are:

- Receive registration request from new parties and perform control of signatures in the request as well as support the administrative process defined.
- Receive list proposals from political parties:
  - Control of candidates against electoral roll (political parties can be given access to this functionality).
  - Inform candidates.
  - Manage information on candidates.
  - Change the sequence of the candidates.
  - Give response to parties (accept or not accept of list proposals).

Configuration of the election(s)

The confirmed parties and candidates that come out of the party registration and nomination process form the basis for the ballots that are to be used in the e-election(s).
The election administrative support system provides the possibility to define and configure one or more election events, specify election milestones etc. Ballots can be configured for different elections. Currently different ballot formats are supported, such as the Norwegian ballot formats for all elections, as well as different ballot styles for referendums such as open questions, multiple choice, etc.

Furthermore, rules for the elections can be defined, such as support for the principle of repeatable ballot casting (i.e. configuring whether or not a voter is allowed to cast multiple ballots, where the last ballot annuls previous ones), whether or not over voting or under voting is allowed, etc

Printing of polling cards
The system produces the necessary data that shall be submitted to a printing central for the generation of a polling card for each voter in the electoral roll. The polling card will hold information about the voter himself as well as all information on the polling stations within the municipality and their opening hours etc. The polling card is also equipped with a barcode that identify the voter and thus simplify the mark on the electoral roll on the Election day.

The security mechanisms that support "the client zero trust issue" are exhaustively described in the security sections of this appendix. The mechanisms used rely on the distribution of some return codes to the voter that can be used to verify that the voter's selections have been recorded as intended. These return codes may be distributed on the polling card to all voters. As an alternative special return code sheets may be made available for those voters that want to do this type of verification process. As the return codes are not tied to a specific person before the actual vote is cast, these sheets may be made available for pick up at some defined outlets.

The system also supports printing of individual polling cards online for stations that are set up for collection of p-votes in advance.

Special Preparation for the pVoting process

Registration and management of sealed ballot boxes (Norwegian: Urner) for storage of votes received in advance
The system supports registration and management of the sealed boxes that keep paper based votes that are received in advance. Each voting envelope received is registered in the system with information on which sealed box it is stored within.

Generation of p-ballot templates
The system generates the templates for all paper-based ballots at the right level (depending on the type of election). These templates obviously include all approved candidates in the defined sequence.

Special Preparation for the eVoting process:

The e-voting solution proposed in this document is based on Scytl's fifteen years of research experience in the electronic voting security field. The solution will reuse certain components of Scytl's groundbreaking product Pnyx in addition to an advanced cryptographic protocol, which will be customized to the requirements of this tender as explain in Appendix 2A Amendment 1. This combination, in addition to physical and logical security measures, provides an electronic voting platform with the highest security levels available today.

Election configuration

During the preparation phase, staff involved with the election, configure the election event using the election administrative support system. Different kinds of elections are supported and can be managed, such as

Municipality, County, General Election, Sami elections, and Svalbard local elections as well as referendums and citizen consultations.

The election can automatically be configured with the approved candidate lists. In addition, it is also possible to manually provide candidates and lists, for example for other types of electoral processes that do not include candidates and lists (such as referendums) or in case the party nomination process is unused. It is also possible to import the ballot definitions.

Rules are defined for the election event, for example if voters are allowed to cast multiple ballots (e.g. the principle of repeatable ballot casting), whether or not over voting or under voting is allowed, etc.

The different election districts, polling places and any relationships between them are defined. Obviously it is possible to reuse such data between different elections.

Electoral Board, election keys generation

Once the election has been configured, the Electoral Board is gathered, and a secure multiparty computation scheme is used to create the election public key and divide the private key into shares without this private key actually ever existing as a whole before, during or after the process. These shares are provided to the Electoral Board members who will keep them under their custody until they are needed after the election to construct the private key for decrypting the votes. A threshold of Electoral Board shares can be defined to construct the key, if this threshold is not meet then the key cannot be constructed and therefore the votes cannot be decrypted, thus ensuring that no single individual can act alone.

The election public key will be used to encrypt each individual vote. As a result, these encrypted votes can only be decrypted with the corresponding election private key which will only be created when it is needed by a threshold of Electoral Board members.

Digitally signing the election

When the election has been configured, the Electoral Board has been constituted and election key have been generated as previously described, the Electoral Board or an administration board on their behalf review and the election configuration. If the configuration is correct, the Electoral Board digitally signs the election configuration and the electoral roll using a distributed signing technique (i.e., with their shares).

The e-voting system validates the signature on the election configuration against the election public key. It is therefore impossible to tamper with the configuration of the election; only the Electoral Board can approve and digitally sign it.

Preparation of the electronic voting platform

Once the Electoral Board has been created, the digitally signed election configuration and electoral roll are securely transferred to the electronic voting platform including the air-gapped Cleansing and Mixing servers. Obviously the integrity and origin of the files are validated by validating the digital signatures. We advise using an offline transfer process where the digitally signed election configuration and electoral roll are burned to a DVD/CDROM. This allows auditors to validate the transfer process. When the electronic voting platform has been prepared, the system is ready to be accessed by the voters when the polls open.

The voting phase

This phase actually consists of two different sub-phases:

- The voting in advance phase
- The election day(s)

For both sub phases it is vital that the system undertakes and supports the controlling role as well as enable clear documentation and audit.

pVoting in advance

The provided solution enables entitled personnel at the municipality to register the advanced votes from their own voters, votes from other communities and advance votes received from abroad. If voters do not have their polling cards, the system provides functionality to print a new polling card locally.

The system supports registration of all votes received in advance. This includes tick off in the electoral roll as well as registration of the ballot envelope itself. For auditing purposes each ballot envelope is connected to the sealed ballot box where it is stored. Entitled personnel at the municipality are given the right to register the votes given in advance.

The votes given in advance are registered in the following categories:

1. In advance votes from voters voting in their “home” municipality:
   a. Votes collected by own staff at defined voting in advance stations.
   b. Votes collected by staff in other municipalities.
   c. Votes collected at institutions.
   d. Votes collected at ambulating polling stations.
   e. Votes received from abroad.
2. In advance votes collected from voters that are registered in other municipalities roll:
   a. Votes collected by own staff (for other municipalities).
   b. Votes collected by own staff in institutions (for other municipalities).
3. In advance votes received from voters that are not in the electoral roll:
   a. Voters that claim to belong to the municipality where they cast their vote.
   b. Voters from abroad that claim to belong to a given municipality. (These votes may be followed by an application for permission to vote. If accepted, the electoral roll is updated with their name and the vote is registered in category 1 e, above.).

The system also supports registration of votes in advance that are rejected. This is necessary for auditing purposes. Any pVote cast by a voter in advance or on Election Day, will annul any eVotes cast by that voter.

eVoting in advance

Voters are allowed to cast electronic votes during the advance voting period. Using a web browser, the voter connects through the voting URL link using a secure channel (e.g., HTTPS) to the voting web page containing the Voting Client which is a small digitally signed Java applet capable of executing the client side of the e-voting solution’s cryptographic protocol. The actual selection of candidates will be carried out using an HTML interface, however the applet is used for authentication and cryptographic purposes.

The voter strongly authenticates by means of a certificate of a trusted certification authority. The planned Norwegian Common Authentication Infrastructure will be integrated into the solution as explained in Use Case 9.1.

Depending on the voter's municipality/county, the correct ballot content is provided to the voter. The voter can be provided with several ballots for different elections and referenda taking place within the context of an election event.

The voter selects the ballot options according to Norwegian election legislation. Before casting the vote, the voter is enabled to verify the vote options and make any changes if desired. When the voter is satisfied with the selected ballot options, she/he casts the electronic vote. The encryption and authentication processes take place inside the java applet within the context of the voter's voting device (e.g. personal computer, mobile phone).

Strong security mechanisms are included to protect the privacy of the vote. A modified and new cryptographic protocol enables the voter to verify that his vote (including all voting options) has been recorded as intended. This feature solves the "zero-trust client" issue.

On the vote collection server, cryptographically chained logs record the main voting actions. This enables audit of all steps in the electronic voting process.

The central electoral roll will be accessed to verify that the voter is eligible to vote and furthermore it is used to mark voters when they cast a ballot. The mark off in the central electoral roll includes whether the vote was casted on paper or it was an e-vote and whether the vote was casted from a controlled environment or not.

The principal of repeated ballot casting

The proposed solution supports the principal of repeatable ballot casting during the advance voting period, a voter can be given the opportunity to repeat the ballot casting process for electronic votes cast from uncontrolled environments. The last vote would annul previous ones. The solution furthermore allows that a voter goes to the polling station on Election Day, or also during the advance voting period, and casts a vote from a controlled environment after having cast one or more electronic votes from uncontrolled environments. This would also annul the previous votes cast by this voter.

Voting at a polling station on the day of election

At the day of election all voters that have casted votes in advance are marked in the electoral roll. As described in the previous section, the mark also indicates whether it was a p- or e-Vote and whether it was cast in a controlled environment or not.

p-Voting

The voter collects her/his ballot and makes corrections if wanted. The voter then identifies herself/himself to the polling station committee. The committee checks the electoral roll. This may be done by a bar-code reader reading the polling card or by finding the voter in the electoral roll if the voter has not brought the polling card.

If the voter is entitled to vote, the electoral roll is updated, the paper ballot is stamped and the ballot is submitted into the ballot box.

The electoral roll will be national and available to all polling stations. This will allow the voters to present themselves at any polling station in the municipality. It must however be decided if the counting still needs to be done at the voting district level. The system supports both registration of votes and counting of votes at the same station as well as registration in one station and counting at another station.

Voters who are already marked in the electoral roll i.e. having submitted p-Votes in advance (and are not entitled to vote according to the rules above) or voters who are not found in the electoral roll, can be registered and allowed to cast their vote. However, these votes are registered and kept separate. The system supports the process to accept/reject such votes (votes in special cover). According to RPA § 10-1 (1) a. – c.

Any p-Vote cast by a voter in advance or on Election Day, will annul any e-Votes cast by that voter.

<u>e-Voting</u>
The system provides functionality for casting of electronic votes on the Election Day (i.e. in a controlled environment). The voting process would be similar to the one described for advance voting. The same technical solution can be used for both electronic voting during the advance voting period as well as for electronic voting on the Election Day.

To cast electronic votes on Election Day at a polling station, standard personal computers could be setup as voting terminals. An election official would validate the identity of the voter in person, and provides the voter with a token (typically a smartcard) which entitles the voter to cast one vote on the polling station. This process would be similar to allowing a voter to cast a paper ballot.

The voter would use this token to activate the voting terminal to cast a vote. It would also be possible to allow voters to use certificates from trusted certification authorities to cast a vote on the Election Day at a polling station.

Any e-Vote cast by a voter on Election Day in a controlled environment, will annul any previous e-Vote cast by that voter.

The counting phase

<u>Tabulation of results</u>

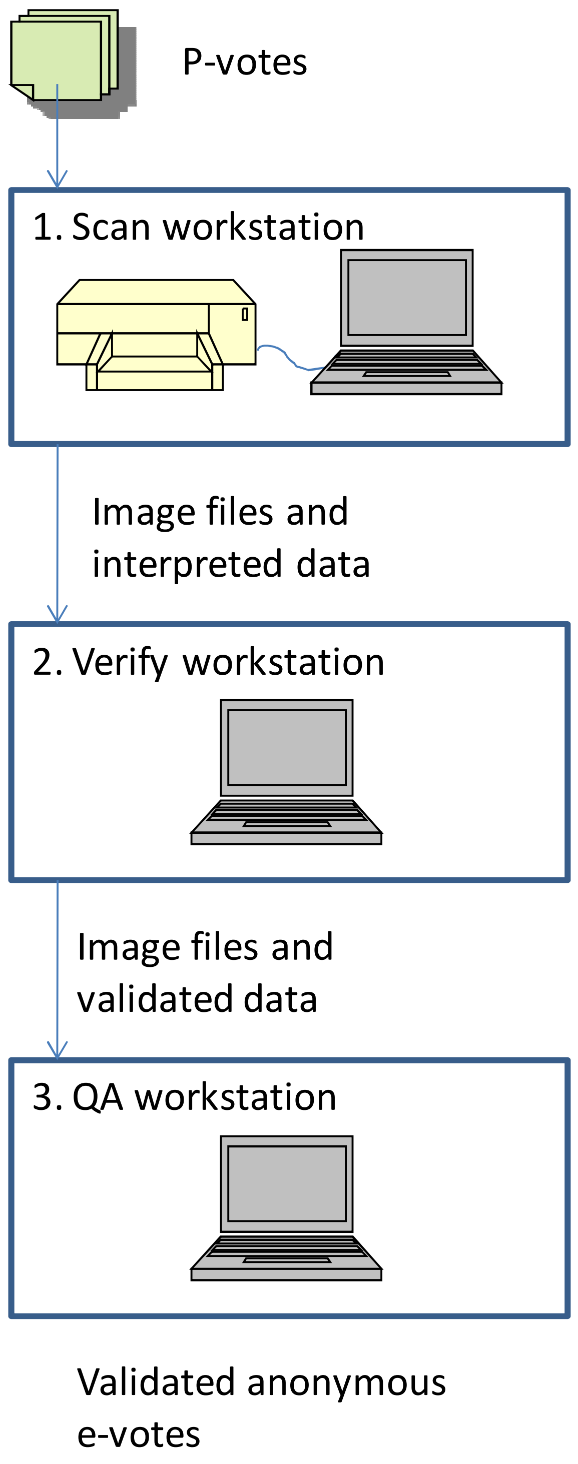
All ballot papers can be tallied in the way the electoral committee decides (each polling station or all polling stations gathered). The ballots can be separated in corrected and uncorrected votes, from voting in advance and from voting on the Election day.

The tabulation system provides functionality to keep tallying results from paper based votes separated from electronic votes (usually for individual statistics purposes).

All the votes can be tallied at the end of the electoral process following the required algorithm, automatically generating several reports about the election results.

<u>pVotes</u>

Electronic counting of pVotes


P-votes
1. Scan workstation
Image files and interpreted data
2. Verify workstation
Image files and validated data
3. QA workstation
Validated anonymous e-votes

Paper votes are categorized according to p-voting categories and location. Batches are produced based on these. The ballots are scanned using a document scanner controlled by an operator using Readsoft Scan software on the PC. OCR is performed real time to extract data from the images. The p-votes are scanned one batch at the time and stay together through the whole process.

Images in TIFF-format and values extracted are transferred to the verify workstation for manual verification

The verify workstation uses Readsoft Verify-software to display uncertain results from the OCR. The operator makes decisions based on directions given by the EC. The EC base the directions on current legislation.

Image files are transferred for electronic storage and the validated data are transferred to the Quality Assurance workstation .

The Quality Assurance workstation use software for validating two separate scans to ensure correct results. The operator can choose to discard results if the two scans differ. If the results from the two scans are equal, the operator accepts the results, and data are transferred to files in the same format as e-votes.

Data in EML-format identical to the data format from the e-voting console are exported, encrypted and transferred to the central collection server. Further processing is equal to e-votes.

Manual counting of pVotes
The results from counting pVotes are registered in the system. In accordance with the administrative regulations, each ballot that is changed by the voter is registered individually and sorted in envelopes. Thus, it is possible to audit each individual vote and verify that the corrections that the voter has made are actually recorded correctly. The system provides both central counting as well as counting at the voting district level.

e-Votes
The e-counting environment is divided into three separate phases: the cleansing phase, the mixing phase and the counting phase.

When the polls close, the digital ballot box and other data collected by the Vote Collection Servers is exported and securely transferred to the e-counting environment. The data downloaded contains the electronic ballot box with the digitally signed encrypted votes and cryptographic proofs generated by the voter. The electronic ballot box is also digitally signed to ensure its authentication and integrity.

The cleansing phase receives the digitally signed ballot box from the Vote Collection Server. The content of the digital ballot box is validated following the established rules as described in requirement F 3.4.1, and produces a cleansed ballot box for each municipality/county (depending on election requirements) defined in the election configuration, as well as the digital receipts from all the e-votes in the digital ballot box that will allow the voters to verify that their vote reached the e-counting process. The digital ballot boxes are digitally signed and made available to the Mixing service. This transmission is done following an air-gapped approach.

In the Mixing Phase, the digital signature of the received ballot boxes is verified to check their authenticity and integrity. The digital signatures of the votes are also verified to check that all of the votes still belong to valid voters (i.e., no rogue votes has been added or modified). After these verifications, the digital signatures from the votes are removed and the votes are processed by a re-encryption universal verifiable Mix-net. This Mix-net breaks any correlation between the votes and their voting order. The output of the Mixing process is the votes re-encrypted and shuffled and a set of cryptographic proofs (ZKP) of correct Mixing behaviour. Once the integrity of the Mixing process has been validated, the encrypted votes are decrypted by the Electoral Board members using their shares of the Election private key (using secure multiparty computation schemes). The list of decrypted votes are digitally signed by the Electoral Board and provided to the counting process grouped by contest.

In the counting phase, votes are counted per candidate/party in order to create the results of the e-votes. This output is digitally signed by the Electoral Board and recorded in a removable device for being transferred to the settlement system.

The system provides publishing of the voting receipts obtained from the e-counting process on a website. Voters can then individually verify whether their voting receipts are included on the published list.

The settlement and reporting phase
The eVote and the pVote tallying results are combined in a secure and reliable way. (This is elaborated in F4.2.1). Thus, both preliminary results and the final tally results consists of all votes collected.

The results for the municipality will be reported to the Electoral committee protocol including statistics.

All required appendixes to the protocol, such as number of votes and the corresponding seats for the lists as well as distribution of mandates, are included. A list that shows which candidates (and alternates) that have been elected as well as number of personal votes is also produced.

Preliminary results
The system has functionality for registration of manual counted pVotes (ballot papers) divided into ballot papers from advance votes (corrected and uncorrected) and ballot papers from the Election day votes (corrected and uncorrected). The results can be registered for each municipality. Electronic counted pVotes from the interface towards the OCR application are imported into the common tally system. The results from the eVoting tallying process are also imported.

The preliminary results may be reported to "SSB" at this stage.

Final tally
The final tally repeats the steps described in the section "Preliminary results" above.

In addition, all ballots that have been corrected by the voter will be processed and any corrections will be registered. For paper based ballots that are counted manually this process will be manual. For corrections done in the eVoting system and ballots counted by the OCR application, the corrections will be registered automatically.

The results from the final tally are reported to "SSB" as required.

Distribution of mandates
The system calculates list-votes for each list based upon the final tally corrected for corrections made by voter on ballot paper. The seats are distributed according to St.Lagüe´s method (modified RPA § 11-4)
Allocation of the seats at large as described in RPA § 11-6 will also be supported by the system.

Candidate declaration
The system will produce a notification letter for each elected candidate and alternates as described in RPA § 11-11 and 11-13. It is also possible to provide an address list (merging document).

Reports
The system proposed by The Contractor will provide the Ministry with a report design and generation tool that can be used by authorized individuals to define, edit or delete, the reports that are used to display the recorded electoral data. This tool is capable of extracting information from the system database (according to the specified permissions) and to create report templates that can be stored on the system for later use. The user will be able to work with reports using a visual interface or in the case of more advanced users by directly using query language. In either case a report template will always have an associated query, an output format and, depending on the nature of the output, it might have a layout (PDF output, for example, will have an associated layout, while CSV format will not).

The reporting tool will have two separate repositories, one for report templates and their corresponding queries and one for the report results (i.e., report templates that have been populated with electoral data and stored). In order to support different types of users such as designers and electoral commission members, user groups and permissions will be supported restricting the actions or information users can see on the system.

After the report templates have been created authorized users will be able to access the tools and select the reports they wish to run. Once selected, the corresponding queries will be launched and the reports will be populated from the electoral data residing on the system. These report results can be further filter, sorted or grouped, if desired, and exported and stored on the user's PC, any external memory devices or repository provided by the reporting tool. Additionally, exported report results can be digitally signed to guarantee their authenticity.

The reporting tool engine will be implemented using JasperReports and the report template generation GUI will be based on iReport.

The reporting engine is independent of the format of the data it will query and the permissions that are applied on that data. When generating a report template, the operator will have to know the structure of the data available for reporting. The data model used for reporting will be documented for the purpose of creating report templates.

The access to the reporting environment requires the user to have a reporting role. However, this does not grant this user unrestricted access to any data source of the election system for making reports. Data source access could require the user to have an additional role for granting access to its stored information and generate reports based them. Reporting system users and roles are managed from the central user/role management system.

This system will, but is not limited to, providing the following general reports:

1. Election Roll
    - 1.1. Election Districts with statistics (Total – men – women)
    - 1.2. List of corrections
2. Parties and candidates
    - 2.1. Participating parties and candidate lists with status
    - 2.2. Approved list proposals with candidates
    - 2.3. List of signatures on list proposals with status
    - 2.4. Letters to candidate
3. Voting in advance
    - 3.1. User defined report advanced votes
    - 3.2. Advanced votes from other municipalities
    - 3.3. Rejected votes
    - 3.4. Statistics
4. Electoral committees
    - 4.1. Number of marks in electoral roll
    - 4.2. Number of ballot papers in ballot box
    - 4.3. Number of votes in cover envelopes

   4.4. Number of electronic votes
5. Election settlement
   5.1. Votes in cover envelop with status
   5.2. Rejected votes in cover envelope
   5.3. Number of ballot papers for each list preliminary counting (advanced voting and election day)
   5.4. Number of electronic votes for each list preliminary counting (advanced voting and election day)
   5.5. Rejected ballot papers (advanced voting and election day)
   5.6. Rejected electronic votes
   5.7. Number of ballot papers for each list final counting (advanced voting and election day)
   5.8. Number of electronic votes for each list final counting (advanced voting and election day)
   5.9. List votes for each lists
   5.10. List votes cast for and received from other lists.
   5.11. List of personal votes (Lists and candidates)
   5.12. Calculations
   5.13. Distribution of seats
   5.14. Elected candidates and alternates
   5.15. Letters to elected candidates and alternates
   5.16. Statistics
   5.17. Election protocol
   5.18. Reporting to SSB
6. Protocol of the polling committee
7. Protocol of the Electoral committee
8. Protocol of the county electoral committee
9. Protocol of the national electoral committee

1.2. Functionality included in the Proposed Solution – free of charge for the Principal

No specific services provided here.

1.3. Other Relevant Functionality Provided

No specific services provided here.

1.4. Other Relevant Services Provided

<u>Option for rental of hardware for full performance test.</u>
The contractor may provide the needed hardware to enable the Customer to run a full performance test. The service offered includes the installation of our software onto the servers at the hosting provider that the Customer selects. Charges are to be agreed upon request.

Option for consultancy from Robert Krimmer
The contractor may provide independent consultancy from Mr. Robert Krimmer, the director and founder of the Center for electronic voting and participation. See attached Robert Krimmers CV. Robert Krimmer is a renowned elections expert offering professional consulting covering the whole electoral process, from the planning and the preparations over the implementation to the independent evaluation and election observation worldwide. In the Norwegian e-voting project Robert Krimmer may take part as an independent consultant supporting and overlooking the deployment, implementation, as well as the actual use of the E-Voting system and the evaluation thereof. Charges are to be agreed upon request.

2. Elaboration of General Requirements

Elaboration of Requirement GR1.3

Our software development methodology and the quality assurance build into the process assure that all system components and modules are fully documented and easily maintainable. The development process and the QA build into the process are elaborated in Appendix 4 – Project and progress plan and in Appendix 5 – Testing and approval.

The development process also includes the necessary effort to prepare for a future Common Criteria certification. One vital part of this is proper and accurate documentation of all modules.

Elaboration of Requirement GR3.10
User training
ErgoGroup can perform training in the Electoral System at.

1. Central level
2. Local level-1 (county)
3. Local level-2 (municipality)

The training will be conducted by the personnel as in previous elections have provided training for Norwegian municipalities in how to use an electoral system and has been involved in the development of the new Electoral System. All training will be given in Norwegian.

User training will include

- System administration
- Overall use of the system
- Use of the Electoral Roll
- Registration of parties and candidates
- Votes in advance
- E-voting
- Counting and registration of p-votes (including scanning of p-votes)
- Settlement
- Reporting

System support
System support will be available working days 08:00-16:00. As support personnel, ErgoGroup is using those who have been working with election support to municipalities in previous elections and has been involved in the development of the new Electoral System. All support personnel speaks Norwegian.

ErgoGroup will assist the Provider in central configuration of the system. To assist the Provider, ErgoGroup will be using those who have designed and developed the Electoral System

3. Elaboration of Use Case Requirements

3.1. Use Case 0.1 Definition of Roles

<u>Elaboration of Requirement F 0.1.1</u>
The system fully provides the management of users. This includes defining users, adding users to roles, suspend users and delete users. By default all users are identified by their social security number which is a unique identifier. All users must be created in advance and named before they are assigned to any role. The system facilitates a web-form where users can request access to certain functionality in the system. The functionality that may be requested by end-users is access to a specific role giving specific functionality. When such access is requested, the owner of the role is automatically notified and must approve the request before the system administrator grants access for the end-user. The approval by the role owner is digitally signed and stored for auditing purposes.

<u>Elaboration of Requirement F 0.1.2</u>
The system provides assignment of users to roles through an approval procedure. It is the owner of the role that must approve the assignment before the system administrator grants access to the role (i.e. include the users unique identifier into the role). Users may be granted to several roles. There is a function in the system that can display all roles one individual user is assigned to. The detailed view of this function displays a complete list of all securable objects that the user is granted access to through the role, including the access rights on each securable object. It is also possible to remove users' granted access.

<u>Elaboration of Requirement F 0.1.3</u>
The system comes with a set of predefined roles. These are:

- Competent electoral authorities, i.e. the electoral board
- Technical operators, supporting the technical system
- Auditor, responsible for auditing the complete election process
- Observer, external observers of the election
- Candidate, candidates for the election
- Voter, the voters themselves
- Independent security body, responsible for verifying the security set up of the system

In addition new roles may be defined based on existing ones or created with a completely new template. The task of creating a new role (in addition to the predefined ones), including assigning the role-owner, must be done by a signed request from the "Competent electoral authorities" and actually implemented by the

"Technical Operator". Thus, two roles/persons are always involved in creating a new role and assigning the role-owner.

Each role has some general attributes that are defined when creating the role:

- The role name and unique ID
- The owner(s) of the role
- The authentication method (i.e. the minimum authentication level required) for members of the role.
- The modification authorization procedure.
    - If the role-owner may add other users to the role on his own, or if collaboration with a System Operator is needed.
    - Which other roles that must be involved when adding or removing access to the securable objects. (The role owner can never add securable objects on his own).

The following are the minimum of operations performed on a role, other necessary operations will also be delivered:

- All management functions such as suspend, delete, update and so-on.
- Add unique users to the role
- Add securable objects that the role may operate on
- Select one of the predefined permission profiles (i.e. permission levels) that this role has on the securable object.

All of the predefined roles in the system, except for the voter role, are mutually exclusive.

Our experience from supporting the voting process for years indicate that the system role "Competent electoral authorities" usually are divided in different sub-roles having different responsibility to facilitate the practical organization of the work at each polling station and in each municipality.

Elaboration of Requirement F 0.1.4
The system has functionality that grants a role access to securable objects. Securable objects are organized into a hierarchical way. Thus, access may be granted at any level in the hierarchy. When the function "grant access to securable object" is activated, a view of the hierarchical structure is shown and a securable object (and its underlying structure) may be assigned to the role.

Elaboration of Requirement F 0.1.5
When the function "Grant access to securable object" is used for a role, and the relevant securable object is added to the role, the different predefined permission levels defined for the securable object are shown. The operator must select one of the predefined permission levels (profiles) that shall apply for that role.

Elaboration of Requirement F 0.1.6
The system fully provides functionality for the definition of access profiles (i.e. permission levels). All access profiles must be defined in advance. There are some system build commonly used access profiles, but others

may be created by the "Technical operator". Valid access profiles that are relevant to the securable object are applied to securable objects by the owner of the securable object itself.

Elaboration of Requirement F 0.1.7

An Authentication and Authorization framework will be used for management of securable objects. The Java Authentication and Authorization service (JAAS) is a potential candidate. The selected framework will provide the ability to integrate with a CAI (National eID) or other CAIs, and an authorization policy database for role based access to securable objects.

Authorization (and authentication) requests will be implemented in every service request requiring access to securable objects. For offline systems (settlement, for instance), a local authentication and authorization service must be implemented with a local copy of the policy database. Exports of the role database will be signed and verifiable similar to other exports from the Election System.

The selected framework will provide functionality to create Hierarchical Role Based Access Control (HRBAC), which then provides permissions based on the role assigned, but may also inherit permissions from roles higher in the hierarchy.

The securable objects themselves will also be placed in a hierarchy. For instance, a screen can have several underlying objects (fields). Thus, the securable objects are organized to reflect typical group of objects that should be managed together by a given role. This best practice ensures that the system may be managed and secured with reasonable resource usage. By using inheritance as the default solution for access profiles, security risks created by unintended changes are minimized.

In this setting, access to securable objects are defined in policies related to the operation the user intends to perform, for instance "update electoral roll." The securable objects affected by the operation will have to established, and what permissions should be granted for each object. However, we don't perceive that there will be a limitation on what the securable object is, rather whether the policy allows the user to manipulate the object.

Access can then be given or rejected with requests at several levels in the code: access to perform the operation as a whole and down to when the securable object is actually manipulated.

The following top branches will be predefined and cannot be altered in the system (roles and objects):

- System (all program components, system configuration files, system keys, logs and so on). The default owner of objects belonging to this branch is the "Technical Operator".
- Electoral role. The default owner of objects belonging to this branch is the "Competent electoral authorities".
- Vote collectors (electronic ballot boxes, scanning system, votes in special covers and so on).
- Results (counting results).
- Settlements.

Candidate securable objects and their granularity

In our elaboration on F.0.1.3 above we have described the different built in system roles in accordance with the "Council of Europe e-vote Recommendation Rec (2004) 11". These roles will be given access policies to the securable objects.

Examples of System securable objects:

- Executable versions of the election system program modules.
- System setup configuration files.
- Any passwords or access tokens to resources that are external to the election program modules.
- Built in root key pair that is used to create other key pairs i.e. signing the pubic key of other keys generated.
- Built in keys that are used by boot-strap program modules to verify the integrity of the system setup.

These securable objects are created by "Technical Operators", are verified and signed by "Independent security body" and are accessed (with read rights only) by the election system program modules.

Examples of Election Configuration securable objects:

- Key pairs used by the different subsystems (i.e. for signing purposes)
- Key pairs for the Election committee(s) (i.e. for signing purposes and for decryption of votes in the allying phase)
- Party lists. They are created by "Candidates" (or the party secretary representing the candidates), signed y the "Competent electoral authorities" and used by the election system program modules.
- Election configuration (i.e. contest, elections and election events). Please note that these objects are at the municipality level. They are signed by the Election committee at the right level (e.g. at the municipality level)
- Electoral roll
- These securable objects are created by program modules or the "Competent electoral authorities", signed by the "Election committee" and used by election program modules.

Examples on securable objects used during election and during the tallying:

- Electronic ballot boxes (to store eVotes)
- System representation of physical ballot boxes
- System representation of envelopes used to store pVotes
- Lists containing pVotes in special cover
- Mark off table in electoral roll
- Scanning result objects
- Voting result objects

These securable objects are created by "Competent electoral authorities" usually as part of the election configuration. The objects are modified (i.e. values are stored) by election program modules or authorized personnel. Some of the object does not accept modification without the signature of one or several authorized persons.

In addition, there are securable objects that hold the settlement result i.e. the distribution of mandates. Immutable logs are also securable objects by definition.

The granularity of the securable objects is regarded with respect to items that need to be handled separately as seen from a security perspective and in the context of the election.

Digital signatures are used extensively to preserve the integrity on and verify the authenticity of securable objects. This is vital as the system itself consists of several components that are air gapped which implies that securable objects are transferred between systems that are not interconnected.

However, access to the securable objects inside each system is controlled within that system (including access to perform the digital signing operation).

Elaboration of Requirement F 0.1.8
There are views in the system that displays the current access rights onto securable objects from two different perspectives:

1. A report may be generated that describes all roles that are granted access to one specific securable object as well as the actual access profile (permission level) for that role. The report includes all inheritance in the hierarchical structure.
2. A report may be generated that describes all securable objects onto which a role is granted access as well as the access profile used. The report includes all inheritance in the hierarchical structure.
3. A similar report as above that display all access rights for a specific individual user. (This report is a combined report that shows the users roles and the access rights onto securable objects for that role).

Elaboration of Requirement F 0.1.9
The results from the report 2 and 3 described in F 0.1.9 preserves the hierarchical structure of securable objects. The different sub-branches may be expanded thus enabling the user to drill down to each specific lower level securable object.

3.2. Use Case 0.2 Configuration of the Election System

Elaboration of Requirement F 0.2.1
An administrator can configure the necessary number of levels, e.g. covering voting districts (bydel) or limited geographical areas for referendums. Normally it will be three configuration levels for the elections:

1. Central level
2. Local level-1 (county)
3. Local level-2 (municipality)

All previous election configuration templates will be available for editing or as a basis for creating a new configuration. An election can be configured without use of previous configuration templates as well.

When an administrator starts the configuration he/she selects:

1. Use an existing configuration
2. Start a new configuration

Elaboration of Requirement F 0.2.2
All previous election configuration templates will be available for editing or as a basis for creating a new configuration. An election can be configured without use of previous configuration templates as well.

If the system administrator wants to create a new configuration, there will be two choices:
1. Use an existing configuration as a template
2. Create a brand new configuration

When an election is configured at Central level, configuration templates will be available for the counties and municipalities at Local level-1 and Local level-2. If other levels are configured, templates will be provided for each level.

Elaboration of Requirement F 0.2.3
The system will guide the system administrator thru the configuration process step by step.

1. The configuration module provides import of election configuration in XML/EML format. It is possible to import either a full or partial configuration. For instance, an XML containing only parties can be imported into a new or existing configuration.

2. The user will be able to set attribute values describing the characteristics of an election/a referendum . All attributes that describe an election will be determined through configuration process. The Election administrator will be guided through the process, so that the configuration contains all the elements it needs. Party codes (and party names) can be imported from different sources. Party codes and party names can be imported from The Brønnøysund Register Center. We recommend "Valgnatt" as the leading source for party codes, but the election system can be defined as the master for party codes. To avoid duplicates, only one system (either Valgnatt or the election system) should be the master for creating new party codes.
   Party codes in ES and "Valgnatt" must be equal to ensure correct reporting to SSB.

3. All language terms will be externalized from the system and replaced with language independent keys/placeholders. These keys will be mapped to the translated terms at runtime. The election configuration module will provide a user interface for creating and maintaining translations, where keys are listed along with corresponding translations for other languages. Terms will be grouped in appropriate modules. Common terms reused across the system will be grouped in a separate module to avoid duplicate entries.

   Export and import of translations will be supported using CSV format. The file has rows with keys and terms. The process for translation is to export terms from an existing language, translate the file and import.

   The admin system supports multiple languages with the assumption that the supported languages are read from left to right.

   The contractor will provide a suggestion of texts for the languages Norwegian (bokmål) and English. The customer will provide further language translations.

Standard software (i.e. iReport) will not support multiple languages, due to challenges when updating to new versions.

4. Roles can be created and maintained on all levels. Central level must create and authorize one administrator on each of the lower levels (19 counties and 430 municipalities). Administrators can create sub-roles on their local level.

5. Creation and maintenance of reports is performed in a separate Reporting module. The reporting tool engine will be implemented using JasperReports and the report template generation GUI will be based on iReport. Reports can be created and executed at any time in this module, also during the election preparation phase.

6. The System will support creation and maintenance of workflows. Workflows will be defined using a predefined set of events and actions. Events are raised by the system when certain incidents occur, for instance when a user is created or modified. Actions define what the system will do when the event occurs, for instance notify an administrator by email. Events can have conditions that are customizable through parameters. The event is only raised if the condition is met. Registration of preliminary results can be such an event, where municipality can be a parameter. In this scenario the workflow will only be initiated when results have been registered for this municipality. Actions are also customizable through parameters. Actions such as notifying by email will have a required recipient parameter that must be defined before the workflow can be stored.

   Orchestration of complex workflows
   Some actions will in turn raise new events, such as requiring approval for a new user. When the user is approved, the event "user was approved" will be raised. All entities requiring approval will follow this scheme. This makes it possible to create complex workflows by chaining different events and actions. For instance, the party list workflow can look like this:

   Event: Party list has been submitted
   Action: Require approval from Electoral Committee

   Event: Party list was approved
   Action: Notify SSB

   Event: Party list was not approved
   Action: Notify party representatives

   Configuring workflows
   The Election System Configuration module will provide an environment for configuring workflows. Users will be guided through all events and can select appropriate actions. The environment will be context-aware: central system administrators will configure workflows on a system and global level, while local administrators will configure workflows relevant for this domain. In the latter scenario the system will provide relevant default values for parameters, such as the current municipality for events that support or require this parameter.

   The configuration module provides visualization of the workflows. Each workflow can be displayed as a model containing the chain of events, actions and parameters. In this model it is also possible to edit parameters for events and actions. Local system administrators will be able to configure all required workflows in this visual mode.

7. Distribution of seats is based on St.Lagüe´s modified method, where the first divisor is 1.4. The divisor 1.4 can be changed by a highly authorized user. The number of seats distributed to each constituency, can be changed. For Parliamentary election the Central level changes the number of representatives and substitutes for each county.

   Configuration parameters such as Election day, other dates given by the "Representation of the People Act", the number of names and signatures on the list proposals and how many corrected ballot papers needed for to be taken in consideration, can be defined by authorized user.

8. Templates for ballot paper layout is created and maintained at central level. On local levels the template can be edited for each participating party/group – i.e. first name/last name or last name/first name, address and occupation.

9. The Election System configuration module contains configuration options for external interfaces and web services. For web services it is possible to edit and validate URLs, configure input parameters, retrieve WSDLs etc. For other interfaces, like file transfer, it is possible to configure authentication parameters, servers and ports etc. The configuration module also provides an interface to the integrated job scheduler. Tasks can be scheduled for execution at a certain time or date, or at regular intervals (for instance each hour). Logs, errors and results from previous executions can be viewed. Tasks that have input parameters can be configured.

10. Export and import of the entire configuration is supported. This export/import will contain both election configuration (type of election, parties, municipalities, rules etc.) and system configuration (users, roles, workflows, translations etc.). EML will be used as format for all entities that it covers. A custom, well-documented XML schema will be used for entities not covered by EML.

Elaboration of Requirement F 0.2.4
When setup is complete, the system administrator can run a test. A configuration test will check for missing configuration parameters and check logical dependencies. A configuration report will be available on screen and for printout. In addition, it will be possible to preview all relevant configurations on screen or as printed material.

Elaboration of Requirement F 0.2.6
When the configuration is completed, an authorized individual is notified and will be able to approve the configuration. Configuration is approved by the administrator or another person with special authorization on each level.

Elaboration of Requirement F 0.2.8
The central/local system administrator must be able to edit an existing configuration up until a preconfigured date. Deadline for configuration will be hierarchic. There will be different deadlines for the different configured levels.

<u>Elaboration of Requirement F 0.2.9</u>
Al previous election configuration templates will be available for editing or as a basis for creating a new configuration.

If the system administrator wants to create a new configuration, based on an existing configuration, there will be two choices

1. Use an existing configuration as a template, with template data
2. Use an existing configuration as a template, without template data

3.3. Use Case 0.3 Electoral Roll

<u>Elaboration of Requirement F 0.3.2</u>
After "skjæringsdato" we will receive the initial Electoral roll data set from SKD. The file may be a sequential file in a prescribed fixed record format or EML. The Election system will load the initial Electoral Roll dataset. The file is loaded into the database system by a batch job customized for this task. We will implement check-sums to control that all records are loaded.

<u>Elaboration of Requirement F 0.3.4</u>
The Election System will receive updates from the Electoral Roll in a predefined time period and intervals configured for the Election/Referendum. Loading updates is performed automatically when the file is received. The Election System will notify the administrator of the result of the update. Receipt will be given if everything is ok, and error list will be produced if there is a deviation. The election administrator in the municipality will be able to create reports of voters who have been inserted, modified or deleted in the Electoral Roll.

If both errors and corrections are transferred in the same file, The Election System will take this into account, and not process errors that will be corrected in a later transaction.

Today the Electoral Roll is controlled by a paper version submitted to public review in each municipality. Public manual control will still be possible. The Electoral Roll for a municipality will be printed from the Electoral System.

We will also develop an online information module for the voters, where voters can verify that they are included in electoral roll. If the voter is not included in Electoral roll and want to complain, the voter can submit a form with an application for correction. The application will automatically be sent to the administrator in the municipality.

When a voter is searching for information about other voters, there will be very limited information that appears. In order to protect the privacy only that person has the right to vote will appear.

<u>Elaboration of Requirement P 0.3.1</u>
The application providing access to the Electoral Roll is expected to be under significant load during peak hours. In order to support 1 million transactions per hour, this application will execute on dedicated application servers and databases. These will be clustered and load balanced. The hardware suggestions for these instances have been made based on this performance requirement; all components selected have proven merits in delivering high performance, scalability and availability.

An important aspect of this requirement is that transactions will be both reads and writes. Import of updates from SKD will consist of writes, performed in batches at predefined intervals. The voters' register for presenting and updating information will be mainly reads, with relatively occasional writes. The database used for this application is selected because it is known to provide very high performance in scenarios with mixed read and write transactions. In order to additionally support this performance requirement, a high-performance disk array has been suggested as storage system.

3.4. Use Case 0.4 Exception Process for Electoral Roll

Elaboration of Requirement F 0.4.1
Voter or party/group: The online information module is used to verify if the voter is included in Electoral roll. If the voter is not included, the voter can fill out a form applying to be included in Electoral Roll. The form is sent electronically to the Electoral System and Electoral Committee is automatically notified that there has been an application.

Electoral Committee: Applications to the election board about a change in the Electoral Roll may be sent by letter. In particular this will be votes in advance from the Norwegians who have lived more than 10 years abroad. It would therefore be possible for an election administrator in the municipality to enter this information manually in the Electoral System. Electoral Committee will automatically be notified that they have received an application.

Elaboration of Requirement F 0.4.4
The Electoral System will have an approval procedure for processing applications. The Electoral Committee will automatically be notified that there has come an application for changes in the Electoral Roll.
The election administrator, who have the appropriate authorization, will be able recall each pending application.
The election administrator sets the proper status of the application, accepted or rejected.

Elaboration of Requirement F 0.4.5
When an application is processed, the electoral register is updated with the results of the application. It will be possible for the Electoral Committee to obtain reports of all changes in the Electoral Roll.

3.5. Use Case 1.1 Submission of List Proposals

Elaboration of Requirement F 1.1.1
Appropriate templates for list proposals will be accessible from web and can be downloaded. The templates will be designed according to the selected election and for use by county or municipality.
Representatives for Parties/groups can as well be given access to the system. Application form for access code will be provided by an external web page.
When the representative for the party/group has gained access to the system, they can select the election/referendum they want to manage candidates/signatures for.

Elaboration of Requirement F 1.1.2
It will be possible to register, edit, import or delete candidates in a preconfigured template. At the municipal council election, extra preference can be given to candidates on top of the list. The number of candidates with extra preference is regulated in § 6-2 (3) and set as one of the national configuration parameters.

The parties/groups type the party´s/group´s name, and the party code will be added during the Election committees processing of proposals.

Scanned files can be attached to a list proposal. A list proposal can be previewed in ballot paper format for the actual municipality/county according to configuration at central and local levels. Withdrawal of list proposals will be possible up to a preconfigured date (§ 6-5) The Electoral Committee will automatically be notified.

The system will perform character recognition (ICR) on signature lists attached to list proposals. After submission of the list proposal, the system will process the list and present the results to the party representative. Signatures that were not recognized must be manually entered. Signatures that were recognized will be presented as text fields that the representative must manually verify. Some characters may have been misinterpreted during the ICR process; these errors must be corrected by the user.

In order to achieve accurate recognition results the system requires that signatures are written on a predefined form. This form should have separate fields with signature, name written with block letters, and social security number.

<u>Elaboration of Requirement F 1.1.4</u>
Each candidate is automatically checked against the Electoral Roll for. The system will return status codes for the candidate:

- Approvable (candidate found in the actual Electoral Roll)
- Approvable after further inquiry (candidates present on other list proposals)
- Approvable with declaration (found in electoral roll, but not in correct municipality/county)
- Not approvable (not found in the Electoral Roll)

Maximum and minimum number of candidates on a list proposal is automatically checked.
For signature lists the status codes will be:

- Approvable
- Not approvable

Number of signatures will be checked according to § 6-3 (preconfigured).

<u>Elaboration of Requirement F 1.1.5</u>

The following checks will be done for each candidate and signature:

- Verify that the signing person is eligible to vote in the actual municipality and county where the list is proposed
- Verify that the signing person is not duplicated within the same list proposal and across list proposals
- Verify that the candidate is not duplicated within the same list proposal and across list proposals

All verifications will be done within the system, and will not be checked towards external registers.

<u>Elaboration of Requirement F 1.1.7</u>
When a list proposal is completed and all candidates are approved by the system, it can be approved by authorized representatives
for party/group before it is submitted to Electoral Committee.
The approval can be done by electronically sign the list proposal or by signing a paper copy which is sent the Electoral Committee by post.

<u>Elaboration of Requirement F 1.1.10</u>
When a listing proposal is approved by authorized representatives of the party / group, stored in the Electoral System and the Election Committee is notified that a list proposal is submitted, the Electoral Committee can publish the list proposals on an external web page or by printing it.

<u>Elaboration of Requirement P 1.1.1</u>
The Election System will be designed and optimized for scalability and high performance. There are several means of achieving this:

- The suggested hardware platform consists of components that are built to scale and perform in high load environments
- The software components that the System uses have been carefully selected, performance merits have been a very important selection criteria
- Horizontal scaling with redundant servers will be used for subsystems that are expected to receive high load. A load balancer will distribute requests across available servers.
- Caching will be introduced at different levels. The Hibernate framework will be used in order to do low level database caching. Front end web servers will cache static content.

The Election System will be capable of delivering the required performance and response time with the indicated amount of parties/groups, counties and municipalities.

3.6. Use Case 1.2 Processing List Proposals

<u>Elaboration of Requirement F 1.2.1</u>
An overview of all list proposals approved by party/groups, and submitted to Election Committee available in the system. Each list proposal can be presented as well.

<u>Elaboration of Requirement F 1.2.2</u>
When Party/Group has completed a list proposal, it is submitted to The Election Committee for approval. The Election Committee can approve or reject the proposal. If the proposal is rejected, the party/group will be notified. The party/group is given the possibility to correct the proposal and resubmit.
When a list proposal is approved, the candidate status is changed from approval pending to approved. Not approvable candidate status is changed from not approvable to rejected.

<u>Elaboration of Requirement F 1.2.5</u>
List over approved list proposals will be exported from the Electoral System to web information pages aimed for the inhabitants. The Election System will interface with external portals for publishing approved parties. Push-based publishing is supported to portals that either provide Web Service interfaces or accept basic HTTP form posts. Party lists will be automatically submitted to such sites as soon as they are approved.

The reporting module can also be used to publish party lists. Reports can be generated that lists parties, and these can be uploaded to public web sites in appropriate formats such as PDF, XML or Excel.

Elaboration of Requirement F 1.2.6
The Election Committee can use the system to run an automatic eligibility control against an Electoral Roll at any time. The Election Committee can edit candidates and signatures on the list proposals until a given date (i.e. deadline for production of ballot papers). A notification will automatically be sent the party/group when a list proposal is edited. The Election Committee performs all corrections after dialogue/negotiations with the party/group are completed.

3.7. Use Case 2.1 E-voting

Elaboration of Requirement F 2.1.1
The Electoral Roll registers the voters and the voting rights for the elections/referendums for which the voters are entitled to vote. The Electoral Roll, the elections/referendums and the voter's voting rights are configured during the Preparation phase.

Once the voter has been authenticated, the system looks up the voter in the Electoral Roll, using the voter's unique identifier (e.g., social security number). The system will not present any elections/referendums to voters who are not registered in the Electoral Roll, who do not have an entry in the Electoral Roll or to voters with an "in-active" status (Norwegian nationals who have lived more than 10 years abroad). Instead, in such scenario the system provides a clear and unambiguous message to the voter with instructions to apply for membership in Electoral Roll.

When the voter is found, the voting rights of this voter are obtained. These voting rights indicate the elections/referendums for which the voter is entitled to vote. If the voter does not have any voting rights, the system does not present any elections/referendums to the voter.

The system provides configuration per election/referendum if it is allowed or not allowed to cast e-Votes on Election Day, from controlled and/or uncontrolled environments. In those scenarios/environments where an e-Vote cannot be cast on Election Day, the system will not present the corresponding election/referendum.

By default, the system will present all elections/referendums for which the voter is entitled to vote to the voter, including those for which the voter already has cast a vote from any environment (e.g. controlled and uncontrolled) and at any specific moment during the vote casting period (i.e. advance voting and Election Day). This helps avoiding coercion or family voting, by not providing any information about any previous cast vote.

In the Electoral Roll it is also indicated if votes were already cast for specific elections/referendums, as well as if those votes were cast from controlled or uncontrolled environments, during the advanced voting period or on Election Day and if the votes cast were p-Votes or e-Votes. Using this information, if necessary, the system can also be configured to filter out and not present specific elections/referendums. For example, not presenting an election/referendum if a voter is casting a vote from a controlled environment, not presenting an election/referendum for which a voter already cast a vote from a controlled environment, etc.

Elaboration of Requirement F 2.1.2
Once the voter is authenticated, the system presents all elections/referendums for which the voter is entitled to cast a vote to the voter. The system allows the voter to select one of these elections/referendums for which he/she wants to cast an e-Vote.

Once the voter has cast an e-Vote for that election/referendum (or cancelled the vote casting session for that election/referendum without casting a vote), the system again presents all elections/referendums for which the voter is entitled to cast a vote to the voter.

The system fully provides the principle of repeatable ballot casting (i.e. voters are allowed to cast any number of e-Votes for any election/referendum for which they are entitled to vote). The voter may repeat selecting and casting an e-Vote for an election/referendum any number of times, without having to re-authenticate. However, if the system is configured to require re-authentication before every contest, every vote or every session (security requirement OS2.1), the voter would be required to re-authenticate once an e-Vote has been cast or a vote casting session was cancelled without casting a vote.

Elaboration of Requirement F 2.1.3
Once the voter has successfully been authenticated and has selected an election/referendum to cast an e-Vote, the correct ballot is presented to the voter. The vote options of the presented ballot depend on the configuration of the chosen election/referendum, which was configured during the Preparation phase. Depending on the chosen election/referendum, the ballot and corresponding vote options presented to the voter may furthermore depend on the voter's voting district (e.g. municipality, county, sami election district).

The table below is an example of configured elections i.e all public Norwegian elections are supported and the vote options for these elections comply with Norwegian legislation. Norwegian Election Law § 7-2 and Regulations for election to the Sami Parliament § 39 (1)

Sample configurations:

Public Norwegian election	Vote options
Municipal Council Election	Give personal votes to candidates in party list. Add candidates/give personal votes to candidates from other party lists. Election Law § 7-2 (2) and § 7-2 (3)
County Council election	Give personal votes to candidates in party list. Election Law § 7-2 (2)
Storting election (parliament)	Change the sequence of candidates in party list. Strike out candidates in party list. Election Law § 7-2 (1)
Sami election	Change the sequence of candidates in party list. FOR 2008-12-19 nr 1480: Regulations for election to the Sami Parliament § 39 (1)

For elections where vote adjustments are allowed, voters are allowed but not required to adjust the vote. Usually, only a smaller percentage of voters make corrections to their ballot. The voter user interface will be designed to accommodate this, speeding up the voting process for those voters that do not wish to make adjustments while at the same time providing full adjustment functionality for those voters who do wish to make adjustments.

For elections that involve selecting a party list, these party lists are directly presented to the voter once the voter has chosen the election, allowing the voter to select the party list of his/her choice. The order in which the party lists are presented is randomized. The system provides two randomization options:

- Complete randomization: the party lists are displayed at completely random positions.
- Cyclic randomization: the party list that is displayed at the first position is chosen completely random but the parties displayed at the subsequent positions maintain the sequence of the original list.

If specific logic exists for the sequence of the party list (for example sorted alphabetically), a cyclic randomization may be preferred over complete randomization for usability purposes as it allows voter to more quickly find a specific party list.

The system provides candidate search if an election is pre-configured to allow for adjustments. This allows a voter to quickly find a specific candidate. The voter can provide characters and the system will list the candidates whose names match the provided characters. This way the voter can quickly find a candidate, selecting from a list and without having to provide the full name. The system provides candidate search within a specific party and also candidate search among all parties in case the voter is unaware which party the candidate belongs to.

The system also provides support for elections/referendums with different kinds of vote options, which are configured during the Preparation phase.

Vote option	Description
Select n out of m	The voter can select one or more answers from a list of questions.
Text input / write-in	The voter can provide free text input (the number of allowed characters can be limited).
Ranked choice / preferential	Rank vote options in the order of preference.
Others	Other less common vote options, such as closed lists or semi-open lists, can be supported if necessary. This will be defined during the project initiation phase and implemented as part of the solution provided.

For elections/referendums where this is allowed, the system provides the voter with the possibility to cast a blank e-Vote.

While the voter has not cast his/her e-Vote, he/she is always allowed to change the vote options. The system warns the voter if any vote adjustments and/or vote options will be lost (e.g., if the voter goes back and selects a different party list, the system warns and after confirmation by the voter, any previous made selections will be automatically cleared). Likewise, while the voter has not cast his/her e-Vote, the voter can cancel the vote casting session for the chosen election/referendum without losing his/her right to vote for that election/referendum.

The exact steps required to provide vote options, fill out a ballot and cast an e-Vote depend on the kind of election/referendum. During the project, The Contractor will work together with usability/accessibility experts/groups to design and implement a user friendly, understandable, usable and accessible voter user interface that allows voters to quickly cast their e-Votes for any of the supported elections/referendums, while enjoying the electronic voting experience.

Referendums
The voter will see the referendum question and will be able to select one of the possible answers in a single screen

Municipality elections
1) Velg Parti:
The voter is presented with the list of parties in random order. The voter will choose the party and be redirected to the chosen parties list of candidates

2) Vis stemmeseddel:
On the screen showing the parties list of candidates the voter can choose to add personal votes or continue without adding personal votes.

3) If adding personal votes, the personal votes page will be presented where the voter can choose as many candidates as he wants.

4) The voter can also choose to add personal votes to candidates of other parties.

5) Bekreft valg:

This page shows a summary of the selected options. If the voter wishes to make corrections, he can go back with the back button or click the “lever stemmeseddel” button to send the vote.

The system provides support for warning the voters to avoid unintentional mistakes and invalid ballot scenarios while providing the vote options of elections/referendums (under-vote or over-vote scenarios, blank vote warning, etc.). The voter user interface will be designed ensuring that the voter cannot make involuntary mistakes while filling out his/her ballot. Depending on election configuration, the voter may or may not be allowed to continue and cast an incorrect vote.

At the Voting Client running on the voter’s voting device, the system can completely validate the votes. Validation is done inside a trusted environment (digitally signed applet) and made against digitally signed election configuration. Where applicable, the following validations are performed:

- Candidates that were added to the ballot are from another party list than the selected one.
- The same candidate is added twice to the same ballot.
- More candidates than allowed are added to the ballot.
- Less candidates than necessary are added to the ballot.

Other validations may be added to meet the required functionality.

Before casting the e-Vote, the system provides a summary screen with an overview of the chosen/provided vote options to the voter. By default, the system presents both the chosen/provided vote options as well as those not chosen/provided. The system can also be configured to only present vote options chosen/provided. At this summary screen, the voter can review his/her chosen/provided vote options. If the voter is not satisfied with his/her vote options, the voter can go back and make modifications as necessary. Once the voter is satisfied, the voter continues to cast the e-Vote.

Elaboration of Requirement F 2.1.4
Before casting the e-Vote, the system provides a summary screen allowing the voter to review his/her chosen/provided vote options. The system provides warnings to the voter to advise the voter about any unintentional mistakes and invalid ballot scenarios. If the voter is satisfied with his/her chosen/provided vote options, the voter confirms to casts their e-Vote.

The Voting Client, a digitally signed applet, executes the client side of the Scytl secure electronic voting protocols before submitting the vote. The vote is encrypted with the election public key whose counterpart, the election private key, does not exist until the Electoral Board creates it. Next, the voter digitally signs the encrypted vote.

The digitally signed encrypted e-Vote is sent to the Vote Collection Server, where the correct construction of the encrypted e-Vote is verified (i.e. verify that the options contained in the encrypted e-Vote are valid ones without disclosing them). Since the Voting Client is executed on the voter’s voting device, any ballot options do not travel unencrypted to the Vote Collection Server. The Scytl secure electronic voting protocols are described in detail in Appendix 2A Amendment 1.

Elaboration of Requirement F 2.1.5
The Vote Collection Server performs a number of validations when it receives an e-Vote:

- The voter who cast the e-Vote must be registered in the electoral roll and be entitled to vote for the election/referendum for which the e-Vote was cast.
- Where applicable, the voter must have sent the correct ballot for the specific election/referendum, depending on the voter’s voting (e.g. municipality, county).
- The vote casting period of the election/referendum for which the e-Vote was received has started and not yet finished. The system supports allowing voters, who initiated a vote casting session before the end of the vote casting period, some extra time to finish the vote casting session. The usage of this feature as well as the duration of the time period is configurable.
- The system supports configuring per election/referendum if it is allowed or not allowed to cast e-Votes on Election Day, from controlled and/or uncontrolled environments. This is validated by the Vote Collection Server.

Furthermore the Vote Collection Server validates the digitally signed encrypted e-Vote, verifying its digital signature to ensure the validity, integrity and authenticity of the digitally signed encrypted e-Vote. The Vote Collection Server executes the server side of the Scytl secure electronic voting protocols, which among others revalidate the voter authentication and the opening token, and ensure the secure storage and processing of the e-Vote. The Vote Collection Server also verifies in Zero Knowledge that the content of the encrypted e-Vote contains valid options (i.e. correct candidate codes for a specific race, or non-duplicated values), and that the vote has not been duplicated using an old session token. The Scytl secure electronic voting protocols are described in detail in Appendix 2A Amendment 1

The validation of the voter digital signature is done through the Central Authentication Infrastructure (CAI) provided by the government for validating eID issued digital signatures. In case CAI only offers authentication services, not signatures, the unavailability of signing services never irreparably breaks the security of our protocol. Assuming that the eID cannot be used to digitally sign the votes (worst case scenario), the “client zero trust” measures implemented by the cryptographic protocol are still effective on detecting any attacks during the vote casting process: the Return Codes do not rely on the validity of the digital signature of the cast vote but on the vote contents.

Cast votes can be digitally signed by an external time stamping authority to preserve their integrity after being accepted. During the whole election, logs are stored in the audit system in an immutable way and therefore, these logs cannot be manipulated to hide any malicious practices based on the lack of voter digital signature. These logs can be used to audit at any time the integrity of the election. Also the voting receipts provided to the voters (in addition to the Return Codes), allow the same voters to verify if their votes are present in the counting process without being modified (i.e., any manipulation of the integrity of the vote does not allow the system to recover the proper receipt identifier).

Our main concern is not related to the security but to the speed and resources required for verifying the integrity of the voting process when digital signatures are not feasible. For example, it is not the same to check the integrity of a vote by searching this in a log register and verify the integrity of the log, than just validate a vote digital signature.

A solution could be to assume in some processes that the previous one properly validated the information provided. For instance, accepting that the a vote collector server digital signature of a ballot box as a warrant that the contents are really votes cast during the voting process. However, we prefer not making these assumptions and check also the validity of the votes contained in the ballot box. To make the process as fast as possible (we know that the speed on having results on time is of paramount importance) without reducing the integrity checks, our solution also incorporates measures for the following scenarios:

1.- Votes cannot be signed using the eID

In case there is no way to use eID for digitally signing the votes, ErgoGroup/Scytl proposal incorporates a key roaming mechanism that allows voters to use digital certificates for signing their votes. This mechanism allows provisioning the digital certificates to the voters in a transparent way (i.e., without requiring to install the digital certificates in the voting terminal). The access to this key roaming mechanism can be based on the eID authentication process or, if eID authentication it is also unfeasible, by means of voter credentials supplied to voters through the voting cards. The key roaming digital certificates are specific for the election, generated during the configuration process and protected using the eID authentication mechanism (or voter credentials). That way, the system could still use the digital signature verification in addition to parallel audits of log information.

2.- Votes can be signed by the eID but not validated through CAI

In this case, we suggest to still digitally sign the votes using the eID. ErgoGroup/Scytl proposal includes local validation of X509 v3 digital certificates based on standard practices (e.g., using trusted CA lists and CRLs). As in key roaming system, immutable logs can be still used to audit in parallel the accuracy of the election. Off course, the measure described in previous scenario is also applicable in this one.

If the validation of the sent e-Vote fails, the system provides a clear and unambiguous message to the voter, explaining the issue(s) that occurred. The voter is informed that no e-Vote has been cast and about possible options to solve the issue(s), if any.

The system makes a preliminary mark off against the voter in the Electoral Roll. Both the electronic voting system and the election administration system have access to a common Electoral Roll. The common Electoral Roll registers among others all voters, the elections for which they are entitled to vote and the mark offs against the Electoral Roll for each cast vote. Access to the common Electoral Roll will be provided following the rules defined in Role Based Security.

The preliminary mark off in the Electoral Roll includes:

- Whether the vote was a p-Vote or an e-Vote.
- If the vote was cast from a controlled or uncontrolled environment.
- If the vote was cast during the advance voting period or on Election Day.

Storing a cast e-Vote and making a preliminary mark off against the voter in the Electoral Roll takes place inside an atomic transaction. In such atomic transaction, a series of operations either all occur, or nothing occurs. A guarantee of atomicity prevents updates to the database occurring only partially.

After the vote casting period, the cleansing process ensures the 1 voter 1 vote principle. During this cleansing process, using the information included in the preliminary mark offs in the Electoral Roll, the system ensures that a vote cast by a voter in a controlled environment for a specific election/referendum will annul/override any e-Votes cast by that voter for that specific election/referendum in an uncontrolled environment. The vote cast from the controlled environment will be the vote that counts. If the voter did not cast a vote from a controlled environment for the specific election/referendum, the last e-Vote that the voter cast from an uncontrolled environment will be the vote that counts.

<u>Elaboration of Requirement F 2.1.6</u>

After successfully storing the cast e-Vote and the vote mark-off in the Electoral Roll, the system provides a clear and unambiguous message to the voter that the vote was cast and the check off in the electoral roll was successful.

This message includes a hash of the encrypted e-Vote and is digitally signed by the Vote Collection Server. The digital signature proofs the integrity and authenticity of the received vote cast notification message and proofs to the voter that his/her e-Vote was successfully cast and stored by the authentic Vote Collection Server.

The hash of the encrypted e-Vote completely guarantees voter privacy and vote secrecy and cannot be used in any way to reveal any of the vote options, chosen/provided by the voter. After the vote casting process, a hash is generated of all encrypted e-Votes that reach the Cleansing process. The system provides publishing a list of all generated hash values to allow the voters to validate that their e-Votes were treated correctly and successfully reached the electoral authorities.

During the project, The Contractor will work together with usability/accessibility experts/groups to design and implement a user friendly, usable and accessible voter user interface for providing the vote cast notification message, ensuring that the hash of the encrypted e-Vote and the digital signature are understandable to voters. The most common alternative would be to provide a barcode representation of these values.

At this stage, voters can also validate that their e-Votes were cast as intended. Together with the polling card, information about the party-lists, candidates and other vote options were sent to the voters. For each voter this includes unique return codes for each party-list, candidate and other vote options. A voter who wishes to validate that their e-Vote was cast as intended can introduce his/her voter code and the system will then provide the return codes corresponding to the party-list, candidates and other vote options what were selected by the voter. This allows the voter to validate that the return codes sent to him/her together with the polling card correspond to the return codes provided by the system. Due that these return codes are generated using secret parameters contained in different entities of the voting system (Vote Collection Service and the Validation Service), as well as the voter code, the voter can then ensure that the contents he/she has casted are the ones that have been received in the system, and that they are the ones that he/she has chosen, allowing the cast as intended verification. The system provides return codes through the voter user interfaces as well as sending them through other channels: SMS or email. In case of using the same channel for return codes as for e-Vote casting, the voter can select some of the characters or coordinates of the return codes to be returned to prevent the impersonation of the service, where these coordinates can be changed at each verification process.

The system calculates the return codes based on the encrypted e-Vote in such a way that the system does not learn anything about the voter choices while calculating these codes. This feature completely guarantees voter privacy and vote secrecy and cannot be used in any way to reveal any of the vote options, chosen/provided by the voter.

The hash of the encrypted e-Votes, the digital signatures generated by the Vote Collection Server as well as the "cast as intended" feature form part of the Scytl secure electronic voting protocols, which are described in more detail in Appendix 2A Amendment 1.

Elaboration of Requirement F 2.1.7

The Electoral Roll registers the voters and the elections/referendums for which they are entitled to vote. Once the voter has been authenticated, the system looks up the voter in the Electoral Roll, using the voter's unique identifier (e.g., social security number).

If the voter is not registered in the Electoral Roll, does not have an entry in the Electoral Roll or has an "in-active" status (Norwegian nationals who have lived more than 10 years abroad), the voter must first apply for membership in the Electoral Roll, before the voter can cast an e-Vote. The system will not allow voters to cast an e-Vote if the voter is not registered in the electoral roll and do not have registered voting rights.

The system will store the exception, including any details about the reason why the voter must apply for membership (e.g. no entry in the Electoral Roll, "in-active" status). The system provides functionality for storing the voter's unique identifier, the date and time the exception occurred as well as any further details about the exception and any additional voter data possibly known at this stage. The system also provides functionality for storing details about the voter authentication process and any certificate that was used. The exception as well as any stored additional information is digitally signed by the system, to guarantee its integrity and authenticity.

The system provides functionality for redirecting the voter to the functionality that handles the exception for voters that are not enlisted in the Electoral Roll, where they can fill in an application for membership in the Electoral Roll.

Elaboration of Requirement F 2.1.8
The Electoral Roll registers the voters and the elections/referendums for which they are entitled to vote. Once the voter has been authenticated, the system looks up the voter in the Electoral Roll, using the voter's unique identifier (e.g., social security number).

If the voter is not registered in the Electoral Roll, does not have an entry in the Electoral Roll or has an "in-active" status, the voter must first apply for membership in the Electoral Roll, before the voter can cast an e-Vote. The system provides a clear and unambiguous message to the voter with instructions to apply for membership in Electoral Roll.

The system provides functionality for redirecting the voter to the functionality for handling voters that are not enlisted in the Electoral Roll, where they can fill in an application for membership in the Electoral Roll.

Elaboration of Requirement P 2.1.1
The system is highly scalable and has very flexible deployment possibilities. It is made up of independent components that only need to be replicated on more servers to be able to support a higher load.

The system follows industry standard architectural design patters and consists of several independent loosely coupled and strong cohesion components. Each system component has its own functionality (service) and can be distributed and replicated among different platforms. The communications among services is based on using fault-tolerant/load-balancing techniques. Therefore, service consumers can automatically switch communications to other redundant services when necessary. Components can be clustered to manage a service for the same group of users without generating inconsistencies (e.g., allowing a voter to vote twice). The architecture is highly scalable and allows adding new components to cover new performance demands or reduce them in case it is necessary, without requiring to shutdown the system.

In its minimal recommended configuration for high availability, the system can support easily more than 30,000 voters per hour. However, in a typical high availability configuration, a layered and well structured infrastructure with redundancy at all layers can support a load considerably superior than the minimum recommended configuration, far over 100,000 votes per hour. The system configuration can be further scaled up to support a much higher load if that would be necessary.

The system design and architecture is further detailed in Appendix 2A Amendment 1

Elaboration of Requirement NF 2.1.1
The system does not provide any information about any previous cast vote at any step in the voting process. For example, the system provides functionality to allow a voter to cast e-Votes for an election/referendum from an uncontrolled environment even if he/she previously cast a vote for that same election from a controlled

environment. These e-Votes cast from uncontrolled environments would simply be ignored during the Cleansing process since the vote cast from the controlled environment is the vote that counts.

Also, there are no features that allow voters to request any kind of information about previous votes during the voting process.

After the election, the system can be configured to publish a list with hash values of all encrypted votes cast. This functionality allows a voter to validate the correct treatment of their votes and ensure that their votes have reached the electoral authorities and the Cleansing process. Following this procedure prevents vote buying since all the valid cast votes are published without distinction between the ones that will count in the final tallying or not. However, this is a system configuration that can be disabled if necessary.

3.8. Use Case 3.1 Registration of p-votes in Electoral Roll

Elaboration of Requirement F 3.1.1
The Polling committee/Election Committee will have access to the Electoral Roll. It is possible to search for a voter in the Electoral Roll by:

- Social security number
- Name
- Date of birth
- Combination of name and date of birth
- Address, e.g. street name or city
- A combination of address and voters name
- The voters number in the Electoral Roll
- Scanning the barcode from election card

The Polling committee/Election Committee can check the following for all voters:

- Whether the voter is listed in the Electoral roll
- Whether the voter is listed in the Electoral Roll for the municipality
- Whether the voter has cast an advanced paper vote
- Whether the voter already has cast a paper vote on Election Day
- Whether the voter already has cast an electronic vote (either in controlled or uncontrolled environment)

A barcode reader device can be attached to the system for reading of barcode from election card.

Elaboration of Requirement F 3.1.2
The Polling committee/Election Committee can mark off voters listed in Electoral Roll for the municipality.
If the voter already is marked off in the Electoral Roll for paper voting (p-vote in advance or p-vote on Election Day) the system will block for further marking. In these cases the system has functionality to register votes in advance or special cover on Election Day.

Elaboration of Requirement F 3.1.4

1. Voters who do not exist in the Electoral Roll shall be given the opportunity to vote in advance or in special cover on Election Day. The system has functionality for registering such votes. All required information about the voter (name and date of birth etc) must be registered in the system manually. There will be a special form for this purpose.

   Advance votes
   A polling card can be printed using the data registered manually.
   Such registration will be stored in the system with status code
   - Advanced vote – not in the Electoral Roll

   Election Day – votes in special cover
   It is possible to print envelopes for votes in special cover envelope - using the data registered manually.
   Such registrations will be stored in the system with status code
   - Vote in special cover – not in The Electoral Roll.

2. Voters who do not exist in the Electoral Roll of the municipally shall be given the opportunity to vote in advance or in special cover on Election Day as described in the "Representation of the People Act" | § 8-4 and § 9-5 (4).

   Advance votes
   Information about the voter will be taken from the National Electoral Roll
   A copy of the polling card can be printed
   Such registrations will be registered in the system with status code
   - Advanced vote – not in the Electoral Roll for the municipality

   The envelope should be dispatched by mail to the correct municipality. The municipality, where the voter is listed, marks off in the Electoral Roll, when they receive the vote.

   Election Day – votes in special cover
   All necessary information about the voter is collected from the National Electoral Roll
   A cover envelope for votes in special cover can be printed
   Such registrations will be stored in the system with status code
   - Vote in special cover – not in the Electoral Roll for the municipality.

3. Voters that have already cast an advance p-vote shall be given the opportunity to vote in advance or in special cover on Election Day**.**

   Advance votes
   If a voter already is marked off in the Electoral Roll for p-vote, he/she will be given the opportunity to cast another vote in advance. The system will notify the operator about previous registrations.
   Information about the voter will be collected from the Electoral Roll
   A copy of the polling card can be printed.
   Such registration will be stored in the system with status code
   - Advanced vote – the voter has already cast an advance p-vote.

It is possible to register "the voter has already cast an advance p-vote" multiple times for each voter.

Election Day – votes in special cover Voters that have already cast an advance p-vote.
If a voter already is marked off in the Electoral Roll for p-vote, the voter will be given the opportunity to cast another vote in advance.
Information about the voter will be collected from the Electoral Roll.
A copy of the polling card can be printed.
Such registrations will be stored in the system with status code:

- Vote in special cover – the voter has already cast an advance p-vote.

<u>Elaboration of Requirement F 3.1.7</u>
A vote in advance is preliminary marked off in the Electoral Roll when it is registered.
When the vote is approved by the Electoral Committee, the mark in the Electoral Roll is changed from preliminary to finally approval.
When a vote in advance is rejected, the system keeps all information and reason for rejection to the vote.
Preliminary marks are not visible in the Election Roll on the Election day.

A vote in special cover is preliminary marked off in the Electoral Roll when it is registered. When the vote is approved by the Electoral Committee, the mark in the Electoral Roll is changed from preliminary to finally.

3.9. Use Case 3.2 Manual registration of p-vote results

<u>Elaboration of Requirement F 3.2.1</u>
All manual registration of vote results can be done through a web interface.
The results can be separated in corrected and uncorrected votes.

Advance results

1. <u>Preliminary results</u>
   Advanced votes can be counted in several rounds according to the "Representation of the Peoples Act" § 10-5(part of the Central configuration) for example on municipal level there will be routines for:
   - Round 1 preliminary count 4 hours before closing of Polling Stations
   - Round 2 preliminary count after closing of Polling Stations
2. <u>Final results</u>
   Results from final count are registered in the same structure as Preliminary count.

Election Day results
On county level the count structure will be on municipalities-level.
On municipal level all ballot papers can be tallied in the way the Electoral Committee decides (each voting district or all voting districts gathered)
Small voting districts can be merged into larger counting districts according to RPA § 10 – 4 2).
Counting structure s is a part of the configuration for Local levels.

The system provides registration routines according to the configured structure:

1. <u>Preliminary results</u>
   Results from preliminary count are registered thru a web interface according to preconfigured structure.
2. <u>Final results</u>
   Results from final count are registered in the same structure as preliminary results.

For all types off election there will be routines for registration of voters' correction on ballot papers. The routines a will accept all legal corrections according to the "Representation of the People Act" § 7-2 (preconfigured at Central level)

It will be possible to import the results in various formats, in accordance with requirement F3.2.1

<u>Elaboration of Requirement F 3.2.2</u>
The system will provide forms for registration of data that is not already stored in the system required for election protocol on all levels.
Registration of data functionality will be provided for the different specified levels (i.e. electoral committee, polling committee and county electoral committee).

<u>Elaboration of Requirement F 3.2.4</u>
The system will present preliminary and final results for the various levels depending on counting structure and election and show differences between those results. Results will be available on screen and as printouts.

<u>Elaboration of Requirement F 3.2.5</u>
Authorized election members will be able to approve the results
The authorization for final approval of results will be attached to election members during the definition of role process.

<u>Elaboration of Requirement F 3.2.8</u>
Authorized persons can edit results for final count. There will be forms in the system for editing the results. All corrections will be logged and will be available for inspection.

3.10. Use Case 3.3 Electronic counting of p-votes

<u>Elaboration of Requirement F 3.3.1</u>
Before start scanning p-votes (ballots) the relevant Committee must decide what is to be scanned (e.g. selection type, location). Each location can be scanned in total or in parts. The "ReadSoft Scan"-software facilitates batch scanning. Each batch is preceded with a form indicating batch identification and location. The form is interpreted by the software and the following images are marked with the batch information as metadata in the ReadSoft database. A batch is terminated by either feeding a "stop-form" or by the introduction of a new "batch form".
The "batch form" can either be preprinted or manually filled out by the operator. If further Meta Data are required, a dialog will be displayed for manually entering at the beginning of each scan.

<u>Elaboration of Requirement F 3.3.2</u>
The recommended scanners produce high quality images in Tagged Image File Format (Tiff). The TIFF image is an industry standard file format and is "platform-independent" (i.e. Windows PC, Apple/Mac, UNIX, etc.

Elaboration of Requirement F 3.3.3
"ReadSoft Scan" has the possibility to OCR the images real-time without loss of scanner speed. This eliminates the use of a post-scanning OCR-engine.

The images are stored on a file share and the metadata and values from the ballot is stored in a central repository (DB). The system is able to recognize minimum 99% of all scanned ballots correctly. If the ballot is unrecognized, it is transferred to a special queue for manual verification on the "Verify workstation".
The system checks that each ballot ID is unique and prompts the user if a duplicate is detected. The operator is prompted on the "Verify workstation". The system can be set up to discard duplicates automatically.

The scanners/pvoting application can read all kind of ballots. Only the party name will be read on ballots from foreign municipalities/counties because of the candidates on these ballots do not belong to the actual municipality/county. If the name of the party/list is written on an ordinary paper, these kinds of ballots can also be scanned but they must be verified manually. The reason is that the application does not know where the party name is written on the ballot paper.

The system is able to scan and interpret special ballots for blind, provided that party information is printed on the ballot. It is important that the ballots do not show if it has been submitted by a visually impaired person or not. This is to support the anonymity of the visually impaired person. This is particularly important on the Election day as it is likely to be few visually impaired voters in each voting district.

Thus, if ballots with Braille signs are used, the ballot should always also contain the same information in regular characters (so that you can never tell if the actual ballot were submitted by a visually impaired person or not). Thus, normal OCR-scanning can always be used to scan even ballots submitted by a visually impaired person person.

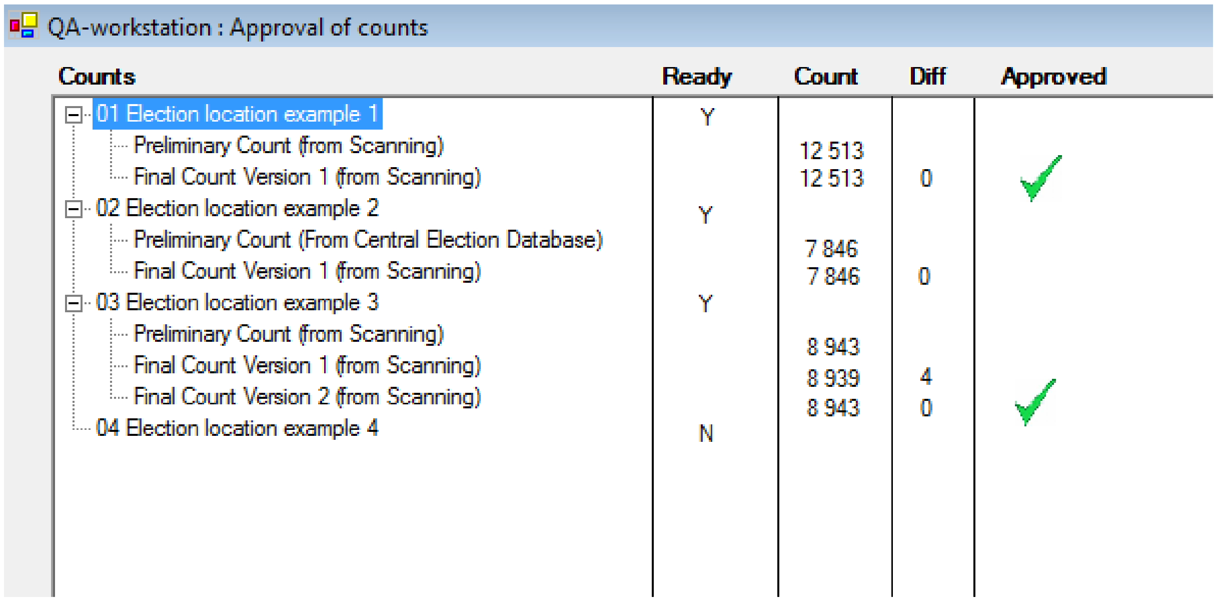
The common way to do this in today´s elections is:

- On the Election day, only the containers holding the ballots have Braille signs. Thus, a visually impaired person can read the front of the container and pick an ordinary character based ballot. However, a better solution may be to also provide the party name in Braille signs as well as the normal character representation on all ballots.
- Advanced votes for visually impaired persons are today based on the general ballot provided by KRD. Today these ballots are used by all voters early in the advanced voting period, regardless of disability.
- Today there are no ballots for visually impaired persons that allow them to add personal votes. This would probably be very expensive as all ballots then would have to be with Braille signs (as mentioned above to preserve the privacy of the visually impaired voter). In this context eVoting with screen reader will probably be the preferred option.

Elaboration of Requirement F 3.3.4
Election results and data for the Election Protocol are stored in the system. The data are viewed and transferred from the QA-workstation.

Elaboration of Requirement F 3.3.5

The QA-workstation presents the results from both preliminary and final counts. If the counts differ, the difference is shown. The information can be viewed as differ in totals, or pr party by double-clicking the count. The counts pr location are displayed by clicking a small "+" to the left of the location name, or hidden by clicking a small "-". In the illustration below, all location counts are displayed.
This illustration can be used as a guideline to the presentation and approval application.


QA-workstation : Approval of counts

Counts	Ready	Count	Diff	Approved
01 Election location example 1	Y			
Preliminary Count (from Scanning)		12 513		
Final Count Version 1 (from Scanning)		12 513	0	✓
02 Election location example 2	Y			
Preliminary Count (From Central Election Database)		7 846		
Final Count Version 1 (from Scanning)		7 846	0	
03 Election location example 3	Y			
Preliminary Count (from Scanning)		8 943		
Final Count Version 1 (from Scanning)		8 939	4	
Final Count Version 2 (from Scanning)		8 943	0	✓
04 Election location example 4	N			

Elaboration of Requirement F 3.3.6

The QA-workstation presents the results from both preliminary and final counts. If the counts are equal, the Relevant Committee can approve the results. See 3.3.5 for an outline displaying the counts. Approval is done by double-clicking the final count that the EC wants to approve. A new window will appear showing a detailed per-party view of the counts. If the final count matches the preliminary count, the EC approves the count by clicking the "Approve"-button (image of approval is not enclosed).

Elaboration of Requirement F 3.3.9

If the counts differ, the Relevant Committee can decide to rescan the votes. The user on the QA-workstation will then be able to choose which version of the counts to compare. It is possible to delete a count by right clicking the count and choosing the option delete. If needed, it is also possible to overwrite manually. Manual changes will be logged and reported on the QA-workstation.

Elaboration of Requirement F 3.3.12

The system will use the "ReadSoft Verify" software to present the EC with rejected ballots. If the ballot is unrecognized by the system, the EC, or the operator appointed by the EC, can approve or reject the ballot according to the current legislation. The ballot will be shown as a high quality image of both sides of the rejected ballot. This can easily be compared to the paper copy. All rejected ballots will be logged and are saved for future reference on the QA-workstation. Approved ballots will be stored according to F3.3.4.
It is possible to open a search interface in "ReadSoft Verify" where the operator can search for specific ballot IDs and display the ballot images. The ballot ID is a unique identifier located on each ballot. This can be used to verify if the ballot has been scanned previously. The purpose of the search is to find ballots that could be true

duplicates, as a result of scanner operator error. If a duplicate ID is found, the ballots in question can be checked to verify if they are indeed the same. To verify identical ballots the changes, stamp and other marks on the ballot is checked.

3.11. Use Case 3.4 Counting of e-votes

Elaboration of Requirement F 3.4.1
The solution provided by The Contractor ensures that the counting of e-votes is carried out with the upmost levels of accuracy and auditability, while guaranteeing voter privacy and vote secrecy at all times. The anonymity of the final decrypted e-votes is completely ensured by cryptographic means, making it impossible to trace a vote to the person who cast it. The following subsections describe the workflow followed by the ballot box after the election is closed.

The e-counting environment is divided into three separate phases: the cleansing phase, the mixing phase and the counting phase.

When the polls close, the digital ballot box and other data collected by the Vote Collection Servers is exported and securely transferred to the e-counting environment. The data downloaded contains the electronic ballot box with the digitally signed encrypted votes and cryptographic proofs generated by the voter. The electronic ballot box is also digitally signed to ensure its authentication and integrity. The exported ballot box has an EML compliant format and is digitally signed following the XMLSig standard.

At the end of the voting period, the Cleansing process receives the digitally signed ballot boxes from the Vote Collection Servers.

The Cleansing process has access to the updated Electoral Roll of the voting process to ensure that e-votes were cast by eligible voters, approved by the Electoral Board. This means that it is impossible to add bogus voters to or to remove eligible voters from the electoral roll. The cleansing process is executed in an air-gapped environment and it can import a digitally signed copy of the Electoral Roll as well as any updates (e.g., voting status changes). Given the flexible nature of the solution offered by The Contractor the Cleansing process can also be configured to be executed on an online environment with access to the online Electoral Roll containing the voting status of all the votes.

Once the votes have been received, the Cleansing process validates the ballot box and the digitally signed encrypted e-votes:

- The digital signature on the ballot box is validated to ensure its integrity and authenticity. If the validation fails, the ballot box is discarded.
- The digital signature on each digitally signed encrypted e-vote is validated to ensure it was signed by an eligible voter. Votes that do not fulfil this requirement are discarded.
- The digital certificate used by the voter is validated (e.g. issued by a trusted CA, trust chain validation, certificate was valid when the vote was cast). Votes that fail this validation are discarded.
- The voter that sent the digitally signed e-vote must be in the electoral roll. If the validation fails, the vote is discarded.
- The voter has the right to vote for the election. If the validation fails, the vote is discarded.

- For provisional votes (e.g. voter applied for membership, trusted certificate without usable voter identification) then only approved provisional votes are accepted. Other provisional votes are discarded.
- The digitally signed e-vote corresponds to the voter's voting district. Otherwise, the vote is discarded.
- The system verifies the voting district that the digitally signed e-vote belongs to and classifies them accordingly. Otherwise, the vote is discarded.

The solution provides offline validation of digital certificates based on X509 v3 standard CRLs (CRL v2). CRLs can be provided as one full revocation list or as delta updates from a starting point CRL introduced when the system is initialized. It could also support verification of any assertion structures (e.g., SAML based assertions) issued by the validation service (e.g., CAI) during an online validation process of the digital signature (e.g., before storing the votes in the ballot box). In this case, the assertion must be annexed to the vote.

It is important to note that as mentioned above, the cleansing process (but not the Mixing and e-Counting process) could be executed in an online environment (e.g., in the vote collector servers). If so, the digital signature verification of the votes can be done online. In this case, there is not a big impact from the security point of view since the digital certificates of the votes are later verified by the Mixing process. We considered in the proposal the cleansing process air-gaped approach based on the requirement of having the Settlement environment air-gaped.

Digital signatures are not only validated until the Cleansing process but also validated before by the Vote Collector Service when votes are stored in the ballot box (see elaboration of Requirement F 2.1.5). This Cleansing process second validation is done to prevent any manipulation of the ballot box from outsider or insider attackers, before accepting the votes for counting (e.g., ballot staffing). Only votes digitally sign by valid voters will be accepted on the e-counting process.

Any attempt to add rogue votes (i.e., votes digitally signed by untrusted digital certificates or voters not belonging to the Electoral Roll) is detected and prevented. We do not assume that the contents of the ballot box are the accurate based on its digital signature. This digital signature is mainly used to verify that the ballot box has not been manipulated after being signed (e.g., during the transport) and comes from a valid source.

Next the Cleansing process will assure the 1 voter 1 vote principle by checking:

- If the voter has cast a p-vote, then any e-votes cast by this voter will be discarded.
- If the voter has cast an e-vote from a controlled environment, then this e-vote will be the e-vote that counts and all other e-votes cast by this voter will be discarded.
- If the voter has only cast multiple e-votes from uncontrolled environments then the last e-vote cast will be the vote that counts and all other e-votes cast by this voter will be discarded.
- If the voter has cast a single e-vote then that will be the e-vote that counts.

At this stage, the Cleansing process has separated all invalid and discarded votes from the valid ones, according to the 1 voter – 1 vote principle.

Finally, the Cleansing process groups the votes in ballot boxes according to the voter constituencies or entity within the administrative level defined in the election configuration. The system can be also configured to join the e-votes from small constituencies together in the same cleansed digital ballot box. This measure helps protecting voter privacy i.e. if only 1 voter in a voting district would cast a vote then by looking at results of that voting district you would know what that voter voted. Finally the cleansed digital ballot boxes are digitally signed and made available to the Mixing service. This transmission is done following an air-gapped approach.

In the Mixing Phase, the digital signature of the received ballot boxes is verified to check their authenticity and integrity. The digital signatures of the votes are also verified to check that all of the votes still belong to valid voters (i.e., no rogue votes has been added or modified). After these verifications, the digital signatures from the votes are removed and the votes are processed by a re-encrypted universal verifiable Mix-net of four (4) nodes. The details of the cryptographic process implemented by this Mix-net and their auditability are explained in Appendix 2A Amendment 1. This Mix-net breaks any correlation between the votes and their voting order. The output of the Mixing process is the votes re-encrypted and shuffled four times (one per Mix-net node) and a set of cryptographic proofs (ZKP) of correct Mixing behaviour. These cryptographic proofs can be used by any auditor, observer or the Electoral Board to verify that the integrity of the votes has not been compromised by the Mixing process. These proofs do not compromise the secrecy of the votes. Once the integrity of the Mixing process has been validated, the encrypted votes are decrypted by the Electoral Board members using their shares of the Election private key (using secure multiparty computation scheme). The decryption process can be based on decrypting each individual vote or groups of votes together (configurable per election). The former approach returns the contents of each individual vote. The later approach returns the contents of the vote set together (i.e., the consolidation of the vote set contents). The list of individual/grouped decrypted votes are digitally signed by the Electoral Board and provided to the counting process grouped by contest.

In the counting phase, votes grouped by contest (or any other electoral division, such as district) are counted per candidate/party in order to create the results of the e-votes. This output is digitally signed by the Electoral Board and recorded in a removable device for being transferred to the settlement system (see Use Case 4.2)]. The results of the Cleansing/Mixing process (digitally signed list of clear text individual/group votes, voting receipts and cryptographic proofs) together with logs, election, configuration and verification data, can be recorded in a WORM device (e.g., CD-ROM) to make this available for audits post election.

The system provides functionality for publishing the voting receipts obtained from the Cleansing Process (as described above) on a website. Voters can then individually verify whether their voting receipts are included on the published list. The presence of a voting receipt on the list of published receipts means that the corresponding encrypted vote reached the e-vote counting process. In addition to this public verification, the receipts of the votes that were decrypted using the Mixing process can be also generated for validation in a controlled environment.

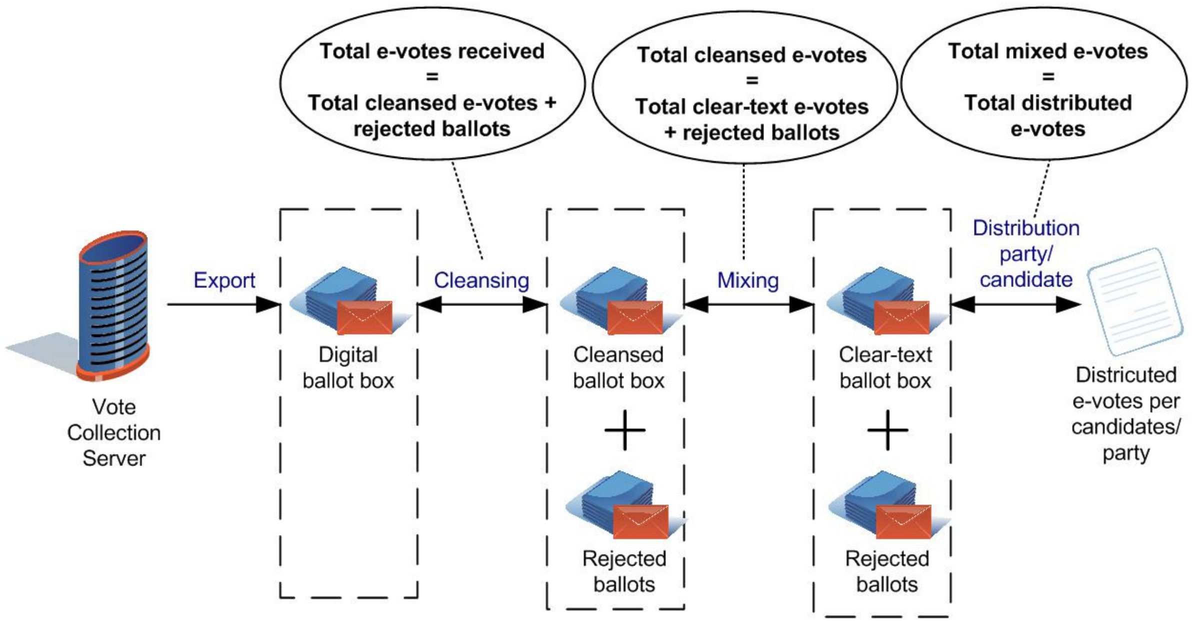

Total e-votes received = Total cleansed e-votes + rejected ballots
Total cleansed e-votes = Total clear-text e-votes + rejected ballots
Total mixed e-votes = Total distributed e-votes
Vote Collection Server
Export
Digital ballot box
Cleansing
Cleansed ballot box
+
Rejected ballots
Mixing
Clear-text ballot box
+
Rejected ballots
Distribution party/ candidate
Districuted e-votes per candidates/ party

Figure 1 – e-vote counting phases.

<u>Elaboration of Requirement P 3.4.1</u>

The main resource consuming process of the e-counting process is the Mixing process. The proposed Mixing process has been designed to be universal verifiable but at the same time optimized for requiring the minimum amount of cryptographic operations: re-encryption of the votes and generation of the group integrity proofs for verifying the accuracy of the project. Details of the optimized cryptographic protocol implemented to achieve this performance can be found in Appendix 2A Amendment 1.

The proposal allows the decryption and counting of 2,000,000 cleansed e-votes in 30 min using 32 CPU cores (i.e., it is feasible using four 2 quad core servers or two 4 quad core servers). The configuration used to achieve this result is based on using a 4 node Mix-net with 2048bits Election public key encryption and group mixing integrity proof checking of 20 votes.

These calculations have been done, taking into account the use of an 'end-to-end' proof approach, where the voter is provided with proof that the cast vote is received, recorded and counted as the voter intended. In addition, the calculations are based on the use of standard off the shelf hardware, and do not contemplate any special components such as cryptographic hardware acceleration.

3.12. Use Case 3.5 Approval of p-votes and ballots

Elaboration of Requirement F 3.5.1
The system will have functionality for registration of p-votes in cover envelope (voting in advance and special votes). The voter will be checked against the electoral roll during the registration process.
Receiving votes in advance and registration of votes in special cover, is usually carried out by election workers who do not have the authority to accept voting. Therefore we will develop a procedure for approval of such votes. The routine in the Electoral System is available for those who are authorized to do so.
The reason for rejecting or accepting votes is configurable. When configuring the election, the actual criteria for rejecting or accepting votes should be defined.

Votes in advance and votes in special cover are today approved by different rules.

Votes in advance
- The system will automatically check whether a voter exists in the Electoral Roll
- The system will automatically check whether a voter has cast a vote several times
- The system automatically checks whether the vote is received by the municipality (registered) at the right time.

All automatic rejection will be listed and can be overridden by the election administrator.
There will also be possible to manually reject or accept a vote in accordance with the requirements of electoral law:
- The envelope has been opened
- It is not possible to identify the voter.

All votes in advance that are not rejected will be given the status of accepted.

Special votes
- The system will automatically check whether a voter exists in the Electoral Roll
- The system will automatically check whether a voter has cast an advance paper vote
  - Prerequisite: The vote is registered in the Election System
- The system will automatically check whether a voter has cast a special vote several times

All automatic rejection will be listed and can be overridden by the election administrator.
There will also be possible to manually reject or accept a vote in accordance with the requirements of electoral law. All votes in advance that are not rejected will be given the status of accepted. All rejected votes, with reason, can be displayed.

Elaboration of Requirement F 3.5.2
After approval, all votes without rejecting code registered in the system will automatically be marked in the Electoral Roll. If a vote on a later stage is rejected, the mark will be removed from the Electoral Roll

Elaboration of Requirement F 3.5.3
The system will store the vote as approved and mark off voter in the Electoral Roll. It is indicated if there is a vote in advance or a vote in a special cover.

Elaboration of Requirement F 3.5.4
All questionable votes must be verified by the Electoral Committee. A form is used to register the number of questionable votes and why they are questionable. The system has functionality (rejecting codes) for rejecting ballot papers according to RPA § 10-3 All rejected votes are stored with reason and can be displayed.

Elaboration of Requirement F 3.5.5
There will be possible manually to register rejected votes in accordance with the requirements of electoral law. Forms for registration will include all the reasons to reject the vote. The reasons are determined by the configuration of the Election System according to the electoral law.

Elaboration of Requirement F 3.5.7
There will be possible to manually register rejected ballots in accordance with the requirements of electoral law. Forms for registration will include all the reasons to reject the ballot. The reasons are determined by the configuration of the Election System according to the electoral law.

3.13. Use Case 4.1 Reporting to SSB

Elaboration of Requirement F 4.1.1
The system has facility to the export data (system data and results) according to formats defined by SSB. It will be possible to report on basic data and election results. It will be possible to report on all levels. The system will present all reports required by SSB.

Elaboration of Requirement F 4.1.2
The system will present all necessary reports for the reporting to the SSB. The reports to be used for reporting are predefined in the system. The Electoral Committee can select the correct report from a contest and run an export of data to SSB. When the correct report is selected, the data to send must be verified of The Electoral Committee before they are sent.

Elaboration of Requirement F 4.1.3
The Election System will transfer data via a web service in the formats defined by SSB. The format is given by SSB before each election. During the configuration of the Election System, the configuration of the reporting is done.

Elaboration of Requirement F 4.1.6
SSB creates a confirmation message which is sent back to the Election System. The Election System will receive and store the confirmation message. The message is stored with timestamp.

Elaboration of Requirement F 4.1.9
SSB creates an exception message which is sent back to the Election System. The Election System will receive and store the exception message. The message is stored with timestamp.

3.14. Use Case 4.2 Settlement

Elaboration of Requirement F 4.2.1
The system is build for maximum security in all aspects. These are described in detail in the section covering the security objectives and in Appendix 3 showing the system design including the different security domains. This section elaborates how the offered solution facilitates a secure and reliable method to aggregate e-votes and p-vote results and to calculate a correct result.

The first interactions between the pVote- and eVote- ballots are in the so-called cleansing process which is part of the counting of eVotes use case. The purpose of the cleansing process is to remove all ballots that should not be counted. The cleansing process take place in an air-gapped environment and is typically performed after the election have been closed and all mark-off of voters in the electoral roll has been done. The cleansing process has a copy of a timestamp electoral roll containing all mark offs. Only those eVotes that have been submitted from voters that has not casted a valid pVote will be forwarded to the decryption and counting process.

After the decryption and counting process of the eVotes (the counting is done at local levels) the counting results are digitally signed by the relevant process in the air-gapped environment. At this stage the results are exported onto the Election Administrative System for statistics purposes and to be part of the preliminary results calculations and reporting.

As described in the elaboration of the use case eCounting of pVotes, the results are digitally signed and exported to the Election Administrative System (EAS) in the same manner as the counting results from the eVotes. In the EAS the content of the digitally signed counting results (including the manually registered counting results) are merged to make preliminary results and reports. It is also possible to do a preliminary calculation for the distribution of seats and returning of members, according to predefined rules.

When all counting results have been recorded all the digitally signed results (both eVotes and pVotes results) are transferred to an air-gapped settlement server that aggregate all votes and calculate the distribution of seats and returning of members, according to predefined rules. These results are digitally signed by the process in this settlement server and exported to the EAS to form the basis for the final reports.

Special note on the Rules & Regulations requirement number 5.1.5 in UC5.1 Reporting:
This requirement states that "T*he system shall prevent the creation of reports where there are less than 100 votes. If there are less than 100 e-votes they have to be merged together with p-votes before reporting from a specific constituency is possible. This will be the same rules as for p-votes with de-central and centralized counting (with reference to the Election Act § 10-4).*"

Even though this requirement has been stated in the use case 5.1 Reporting, the solution to this requirement is implemented in the settlement process. At all levels the counting is performed there at stored three values: Number of eVotes, number of pVotes and number of aggregated votes. As described in use case 01. Definition of Roles (and securable objects) these numbers are controlled. If the number of Votes at one level is less than 100, the object will not be acceptable for other than the owner of that objects. Thus, this number will not be possible to include in reports.

Elaboration of Requirement F 4.2.2
When all counting results have been recorded all the digitally signed results (both eVotes and pVotes results) are transferred to an air-gapped settlement server that aggregate all votes and calculate the distribution of seats and returning of members, according to predefined rules. These results are digitally signed by the process in this settlement server and exported to the EAS to form the basis for the final reports. The system fully fulfills the St.Lagüe´s modified method currently used.

3.15. Use Case 5.1 Reporting

Elaboration of Requirement F 5.1.1
When an authorized user accesses the system he/she will be able to browse through a list of existing elections/referendums that are on the system. By selecting an election/referendum he/she will be able to see all the reports associated to it, and according to the users specific permissions, edit, delete or create a new one. It is important to note that report templates can also be defined by election type so that they appear for all the election of that specific type (e.g., a report appears on all the municipal elections on the system).

Elaboration of Requirement F 5.1.2
The reporting tool engine will be implemented using JasperReports and the report template generation GUI will be based on iReport.

JasperReports is an extremely flexible open source reporting library, which includes various advanced features such as, but not limited to, the ones below:

- Generation of documents in a wide range of formats. Please refer F 5.1.8 for a complete list.
- Report definitions that include charts layouts, such as Pie, Bar, Stacked Bar, Line, Area, Scatter Plot, Bubble, and Time series.
- Multiple sources can be merged together. The data can be retrieved from defined data sources such as JDBC, CALS Table Models, JavaBeans, EJBQL, XML, Hibernate, and Comma-separated values, and additional data sources can be added to the JasperReports.
- The use of Sub-reports.

iReport is an open source standalone graphical program that provides report designer capabilities, and is able to run reports using all data source supported by the JasperReports engine.

New report templates can be created and stored on the system at any moment before/after/during the election. If the report designer decides to use the GUI report designer, iReport will be started. Otherwise, the XML report language can be used to create a new template if a non visual environment is preferred. Using the iReport GUI, authorized non-developer users will be able to generate reports based on the system tables. Once the report template is stored in the system, it can be executed at any time by users with the correct permissions.

Elaboration of Requirement F 5.1.3
The reporting tool will provide the designer with an option to preview the report on screen at any time during or after its configuration. This preview will allow the designer to verify that the results are as desired. The preview can be closed at anytime in order to continue working with the report.

Alternatively a printed preview of the report will also be available, in order to allow the design to generate a hardcopy example for testing purposes.

Elaboration of Requirement F 5.1.4
When a report template is created it can be stored on the system for later execution or modification. When creating the report the designer will be asked which election/election type he/she wishes to associate it to.

Elaboration of Requirement F 5.1.5
The Electoral Committee will be able to access the reporting tool and select election/referendums from the list that will appear on screen. Once the election has been selected a list of reports that are available for the election/referendum in question will be displayed.
Regarding the non functional requirement rules and regulations 5.1.5, the control to prevent queries of less than 100 votes will be present in the settlement system described in Use Case 4.2 and therefore there will be no need to control this requirement form the reporting tool.

Elaboration of Requirement F 5.1.6
The Electoral Committee will see a list of all the available report templates stored within the system for each election/referendum. The election/referendum is selected as described in F5.1.5. The list of available report templates will be displayed on screen. By selection a report it will be run by the system.

Elaboration of Requirement F 5.1.7
After selecting the necessary report template in F 5.1.6, to the system will execute the query that populates the report and shows the results on screen.

Elaboration of Requirement F 5.1.8
With the report results selected in F 5.1.8 on screen, the Electoral Committee will be able to sort, filter and group the data as required using an extremely user friendly interface. Once the report set has been changed and executed, the JasperReport engine will be invoked with the selected report template and the filtered and grouped query will be executed. The Electoral Committee will be able to store those results in several formats, the most common being: CSV, HTML, XML, EML, PDF, but others commonly used publishing formats are also supported.

The results of report execution can be stored in any available location (local or external disk) on the computer as well as on the system. The repository of report templates will be separated from the repository of report results. When results are stored as a file, this file can be sent as an attachment to any e-mail.
When the results of a report execution are stored in the system, it will be possible for users that are authorized to review the results and set a mark of acceptance of the results, as a result these reports can be digitally signed so as to indicate that it has been approved. Thus supporting election protocol in regards to report approval procedures.

As introduced in the elaboration of requirement F 5.1.2, the reporting tool engine will be implemented using JasperReports and the report template generation GUI will be based on iReport. Both are open source solutions that can be used from heterogeneous platforms (e.g., Windows, Linux, MacOS, etc.) to create any kind of reports. These reports can be shown on the screen, send to printers and recorded using different file formats including, but not limited to, the ones demanded by the requirements: CSV, HTML, XML, PDF, RTF, ODF, OOXML and image formats (BMP, GIF, JPEG, PNG, TIFF, EMF).

Jasper Reports provides recording OOXML formatted files by means of using the Apache POI Java library. Documents generated with this format are also editable, as well as in the other formats.

EML format support is not native but will be provided by ErgoGroup/Scytl as a Jasper Report plug-in developed for this purpose. Report templates based on the different EML schemes will be also provided as reference.

Since Jasper Reports has a very active open source development community, new plug-ins can be added or developed in a future without requiring any license.

Regarding the support of formats required for the production of ICR readable ballots, the production of polling cards, or commonly used publishing formats, the proposed reporting tool is very flexible on defining formats and gathering information from different data sources:

- The iReport GUI previews the result of the document template before generating and printing/recording the ballots, polling cards or any other report. Layout can be changed and pre-viewed again as many times as required. Multiples fonts and font sizes are available to adjust the document layout to the requirements for producing any kind of ICR readable ballot. The final version of the report template can be stored for sharing or re-using it in a future.
- The reporting tool can access different data sources for gathering the information required for generating the reports/documents. For instance, the reporting tool can access the Electoral Roll information to gather the voter details required to personalize the vote cards. Using a voting template the generation of the reports can be automated (i.e., printed or stored in files based on the above mentioned formats). The same process can be used for the generation of ballots or reports.
- The reporting tool includes automatic graphic chart generation based on the information gathered from the data sources. Graphic charts are embedded in the printed/stored reports. It provides multiple chart layouts, such as Pie, Bar, Stacked Bar, Line, Area, Scatter Plot, Bubble, and Time series. These charts can be used for instance in the result reports (e.g., HTML formatted results for web publishing or PDF result files for sharing results with media).
- The reporting tool also provides the generation of barcode images for reports (e.g., vote cards). These barcodes can be dynamically generated using the information of a data source. The tools can generate linear (1D) and matrix (2D) barcodes.

Elaboration of Requirement F 5.1.9
After a report template is stored, it can later be selected/modified at any time. The report designer will have a list of stored report templates by election/referendum, from which he can pick the one for revision/modification.

Elaboration of Requirement F 5.1.10
After a report template is stored, it can later be modified at any time. Modifications done can be applied as changes to the current report or stored as a new report template. This will allow creating new templates based on existing ones, saving significant time. Note that editing a template will not modify the report results based on the template in question that are already stored on the system, just the ones that will be executed using it in the future.

3.16. Use Case 5.2 Auditing

Elaboration of Requirement F 5.2.1
The system logs all significant events, recording among others user, time and event details. This includes logs of all events at all levels of the complete Election System, including all election transactions, Attacks on the operation of the election system and its communications infrastructure, System failures, malfunctions and other threats to the system and events at operating system level. Please refer to elaboration of OS13.3 for further elaboration on event logging.

Log messages recorded are status/informational messages (i.e., executed transactions and their result) as well as errors/issues. All log entries contain the following information:

- Timestamp: using the clock of the computer hosting the log originator.
- Origin: the service originating the log.
- User/voter information.
- Type: error, status, information.
- Message/event details.

Format of log entries and additional detailed log information can be further elaborated in the design phase as needed to satisfy the requirement for auditing.

Election transactions
The system stores all election transactions in immutable logs. The logged election transactions allow implementing a complete, meaningful and voter privacy compliant audit of the election, from beginning to end. Even though the elections transactions allow auditors to completely trace the election, including voting transactions, they cannot compromise voter privacy, the integrity of the election or the integrity of any audit information. Immutable logs only record the encrypted and digitally signed votes. Voting receipts do not discern the intention of vote. The Mixing process provides the list of decrypted votes and voting receipt identifiers but in a complete different order than the encrypted votes. All audit information is at least digitally signed (logs are also cryptographically chained) to prevent the manipulation by auditors. Election configuration, logs, audit information, etc. can be stored on WORM devices. Auditors do not require private keys to audit the information (i.e., only public keys are required to validate the digital signatures)..

Election transactions are logged for (not exhaustive):

- Stop/start-up component and service information.
- Election configurations.
- Election import events.
- Electoral roll import events and the number of eligible voters.
- Election stop/start/end transactions.
- Election import/export transactions.
- Ballot box export/import events.
- Voter authentication transactions.
- Vote casting transactions.
- The number of votes cast.
- The number of invalid votes.
- Cleansing process events, transactions, results and further audit information.
- Mixing process events, transactions and results and further audit information.

- Distribute/Count e-votes process events, transactions and results and further audit information.
- The counts and recounts at any necessary administrative levels.
- Detailed information to allow reconciliation of different processes at necessary administrative levels.
- All creation of securable objects and changes to their access profiles.
- All creation or changes to role definitions.
- Any transaction that add or remove persons from a role.
- All state changes on election configuration objects, including creation, submission for review, submission for authorization, authorized, changed, made obsolete and so on.
- All entry of counting results and the signature/approval events.
- All manually registered votes and results.
- All key generation events and all management operations on keys.
- Any errors/issues that occur.

Attacks on the operation of the election system and its communications infrastructure
Suspicious events and attacks are recorded at two levels:
- At solution component/service level (e.g., repeated invalid authentication attempts, digital signature validation failures, etc.).
- At technical level, based on the operating system, web server, web application, firewall (local and perimeter), IDS or any other system or infrastructure component log information.

System failures, malfunctions and other threats to the system
This information is also are recorded at two levels:
- At solution component/service level.
- At technical level, based on the operating system, web server, web application, firewall (local and perimeter), IDS or any other system or infrastructure component log information.

The audit logs are protected against unauthorized modification
The audit logs are protected against unauthorized modification. The immutable logs protect the integrity and authenticity of all the election transactions registered by the solution. Immutable logs are based on cryptographically chaining the log entries and making digital signature checkpoints of these chained entries. Therefore, in case a single log entry has been manipulated (i.e., illegally added, changed or deleted) this change is located and isolated from the rest. This is a great improvement compared to only digitally signing a complete log file, since in this case any modification would invalidate the complete log, without having the possibility to detect where the manipulation is done. Furthermore, it provides a meaningful audit, since in addition to detect manipulations to the audit logs, it allows discerning which parts of the log entries are valid or invalid for the audit. Immutable log technology is not only used to protect all election transactions, but also to protect log messages/events generated at operating system and standard infrastructure component level (through syslog functionality). Immutable log technology forms are further described in detail in Appendix 2A Amendment 1.

Access to all logs stored on the system is furthermore controlled through strict access control, procedures and processes. Physical access to the election platform infrastructure is also controlled. Logs are stored on WORM devices.

Elaboration of Requirement F 5.2.2
Auditors can be provided access to all audit logs before, during and after the election. This allows the auditors to conduct a meaningful audit, at any desired moment.

The system provides a user interface for auditors that they can use to filter and search through the logs. Extended filter and search criteria are available to facilitate audits (not exhaustive):

- Log entries generated between date/time ranges.
- Service/component that generated the log entries.
- Trace log entries of specific users/voters.
- Status/error types.
- Integrity status (e.g. broken chain, invalid signatures)

The auditor can furthermore use the report generator to search through the audit logs and provide very specific access and filter criteria.

The format of the audit logs are based on text files allowing the auditors can also use their own tools if they would prefer such approach. Documentation about audit log entries will be provided.

The solution includes a web dashboard that allows election officials or auditors to follow up the status of the election. It shows real time election status information such as: the status of the voting process (open/closed), the number of eligible voters, the number of successful/unsuccessful login attempts, the number of votes cast, the number of voters that has participated, etc.

The Cleansing, Mixing and Count e-votes processes also store detailed log information in immutable logs and provide on the screen the information of the processed votes, including but not limited to: the ballot boxes imported and the owners, the spoiled votes and their reason (duplicate votes, votes that has a p-votes, etc.), the number of invalid votes and the reason of the invalidation, the counts/recounts number and results, the voting receipts recovered, etc.

Elaboration of Requirement F 5.2.3
All the log information mentioned in this Use Case will be stored on a database. This information will be available for creating detailed reports, using the reporting tool described in Use Case 5.1

Elaboration of Requirement F 5.2.4
Every operating system, application, and device on the network generates log events to inform about current system status. The system collects, analyzes and correlates these logs to raise an alert in cause of attacks or errors.

The system has a set of agents (log analyzers) which are responsible for analyzing the different logs. Each agent is designed to work with a specific kind of logs: firewall, IDS, voting systems, etc.
Agents can be installed on the same server they are monitoring or installed in a centralized one in order to remove the load of analyzing logs from specific platform servers.

Elaboration of Requirement F 5.2.5
The system proposed by The Contractor allows the Electoral Authorities to configure which incidents will raise an alert, allowing them to distinguish critical incidents over the regular noise on any system. Integration with SMTP, SMS, and SYSLOG allows the auditors to receive alerts through various channels, such as e-mail and handheld devices (e.g., cell phones and pagers). Every system agent allows configuring the events that raise alerts.

In addition, the system can integrate with current systems such as SIM/SEM (Security Incident Management/Security Events Management) products for centralized reporting and correlation of events.

Elaboration of Requirement F 5.2.6
The system proposed by The Contractor provides a simplified centralized management server to manage policies across multiple operating systems. This management service will allow auditors to define server specific overrides for finer grained policies.

However, depending on the nature of a specific agent, its configuration may vary significantly. The steps needed to configure an IDS log agent are very different than those for monitoring a Web Server. As a result, each agent will have to be configured separately. However the system will provide the means for auditors to carry out the configuration of the parameters, without interfering with the electoral process, and guaranteeing voter privacy and vote secrecy at all times.

Elaboration of Requirement P 5.2.1
Logs and alerts will be stored on a dedicated system, removing the load from the voting servers. All the log auditing tasks will also take place at the centralized server. The agents installed in the voting servers will be very light and will not interfere with the voting server performance or response times experienced by voters.

Elaboration of Requirement P 5.2.2
All logged events will be store on an indexed database for querying and on files for signature verification. This guarantees a low response time when running log queries, which in most cases will be less than 1 second. This of course will vary depending on the detail of the logs and parameters used to configure them by auditors as well as database performance.

3.17. Use Case 9.1 Authentication

Elaboration of Requirement F 9.1.3
The proposed solution provides several authentication schemes for inside and outside users. From the architectural point of view, the authentication is based on an independent module, facilitating the integration with different authentication schemes without requiring modifying the complete solution.

Authentication is managed at two different levels: outside users (voters, party secretaries and so on) and inside users (Election managers, operators, Electoral Board, etc.).

Outside users:

The outside user authentication method is based on strong authentication mechanisms (i.e., digital certificates). The system will integrate with the national Common Authentication Infrastructure as required.

Inside users:

Inside users are those users that will have some kind of privileged access to the election management or voting system. The authentication methods supported for these users are based on strong authentication mechanisms (i.e., digital certificates) for critical processes or other mechanisms (e.g., username/password) for less critical processes or where strong authentication cannot be implemented (e.g. air gapping security mechanisms). In addition to single user authentication mechanisms, the solution also implements cryptographic mechanisms that require the collaboration of a threshold of users to access to highly restricted election objects.

Elaboration of Requirement F 9.1.9
The system will after having established the identity of the end user (by proper authentication), access the authorization system to determine which roles that the user is member of.

Elaboration of Requirement F 9.1.10
The roles (if more than one) will be displayed to the user and the system will wait for the user's selection. The chosen role will be sent to the Election System and will be used by the authorization system to discern the user's access to any service or object. By default the selection of one role is valid for the complete session. Thus, normally a new logon/authentication is required to select another role.

Elaboration of Requirement F 9.1.11
The system provides temporary one-time credentials that may be used for voters that cannot present an approved e-ID on the Election Day and want to cast an eVote. In case the voter is eligible to vote, the election official can assign a valid credential to the voter to cast a vote in one of the voting terminals available in the controlled environment. This credential will be based on a key roaming mechanism and therefore, the voter will be assigned with a unique digital certificate for casting the vote. This credential can be stored on a cryptographic token (e.g., smartcard) that must be introduced in the voting terminal. It can be also provided inside a sealed PIN tamper evident envelop. In the latter case, the voter should verify the integrity of this envelop before opening it, and introducing the credential in the voting terminal. The approach based on smartcards would be the one that is recommended to use with voting terminals in controlled environments.

Elaboration of Requirement F 9.1.12
When the voter accesses the polling station in the controlled environment, he is given the option to use an approved e-ID or, in case the voter arrived at the Polling Station without an approved e-ID, to use the temporary credentials that are supplied by the election official (after having controlled the identity of the voter and that the voter is eligible to vote).

Elaboration of Requirement F 9.1.13
After having selected to use the temporary credential option the user "enter" the credentials. This is either done by inserting the SMART-card that holds the time limited one time token or by providing the PIN in the sealed tamper evident envelope (which alternatively could be scanned as a barcode), if that option is selected. The generation of the one-time PIN will be part of the system functionality, but production and distribution cost for PIN envelopes is not included.

Elaboration of Requirement F 9.1.14
Use case 9.1 Authentication, section 7.2 (which this requirement applies to) defines the alternative flow for VOTERS with temporary credentials. In this case the voter is authenticated by the election official before the actual eVoting is performed. As described, a key is generated and a one-time certificate is provided to the voter, through a key-roaming feature. This certificate is used for authentication of the user at the polling station, thus enabling him to cast his vote.

4. Elaboration of Accessibility and Usability Requirements

Elaboration of Requirement AU 1
The front-end both for the voting application and that administration module is developed using hand-coded HTML styled with CSS. Every care is being taken to ensure that the base HTML follows a logical flow. It is and will be during the development phase proper, continuously tested with the browsers listed below.

Elaboration of Requirement AU2
Since we are in full control over the front-end using HTML and CSS, the e-voting client will work for and be tested against all major browsers, including text-only browsers and screen readers. Below different alternatives are described. The final decision will be taken in the detailed design phase. The actual alternative selected will not influence on the pricing.

The current requirements in the client side, in addition to an HTML browser, are Java and JavaScript. This current requirement only limits the browser support to those that can execute Java applets and JavaScript.

Current OS and browsers combinations that may use the default proposed e-voting client (i.e., JavaScript and java applet combination) can be summarized in the following table.

Browser support for **eVoting** client:

	IE 6	IE 7	IE 8	FF 3.5	FF 3	O 9	O 10	S 3	S 4	Ch	K
Windows XP	x	x	x	x	x	x	x	x	x	x	
Windows Vista (32/64)		x	x	x	x	x	x	x	x	x	
Windows 7 (32/64)			x	x	x	x	x	x	x	x	
Linux (32/64)				x	x	x	x				x
OS X 10.4				x	x	x		x		x	
OS X 10.5				x	x	x	x			x	
OS X 10.6				x			x			x	
Solaris 10				x	x	x	x				x

IE: Internet Explorer
FF: Firefox
O: Opera
S: Safari
Ch: Chrome
K: Konqueror

However, there are other configurations (pure Java, pure JavaScript and server side voting client) that can be supported in the final implementation.

A full Java based voting client solution (e.g., the Java applet solution provided in the first prototype) can be implemented as a fallback solution without JavaScript support. This solution can be executed at the voter terminal or in an external server. In the case of using the voter terminal, there are two possible approaches: as a Java applet embedded in a web page or as a Java jump-start application. In the case of using an external server, the application is implemented as a Java servlet and therefore, the voter interface could be based on any pure

HTML browser (i.e., accessible by any browser, even text-based). The drawback of this latter approach is that the voting process is implemented in a server instead of the voter terminal (i.e., raises privacy concerns). However, vote integrity is still ensured by means of the return codes (i.e., the server system cannot modify the voter intent without being detected).

It also is feasible to implement the applet functionality using a non-interaction JavaScript (there are "big number" JavaScript libraries that allow this implementation). However, it is important to note that JavaScript interpreters of browsers are not always using the same JavaScript implementation and should require to develop different JavaScript voting client versions based on the browser.

Pure JavaScript implementation is supported in two ways:

- Using the Java Runtime environment of the system: this implementation does not use a Java applet but still requires a Java virtual machine installed in the voter platform. In this approach, JavaScript makes direct calls to Java runtime libraries for executing certain operations (e.g., generation of cryptographically secure random numbers). The drawback of this approach is that still requires to have a Java Virtual machine installed in the voter terminal but reduces the impact of JavaScript incompatibilities among browser versions.
- Using pure JavaScript approach: in this case, special libraries focused on managing "bignumbers" and other cryptographic operations are required. The advantage is that it does not require a Java virtual machine installed in the voter terminal. However, the drawback is related to support multiple JavaScript browser versions and the performance of the solution (JavaScript was not initially designed to implement complex operations as Java does).

In case the reason for using JavaScript is to prevent showing the notification screen that appears when an applet is signed, it is also feasible to implement the voting client using a non-digitally signed Java applet (JavaScript cannot be signed and therefore, an unsigned java applet is not less secure than using JavaScript).

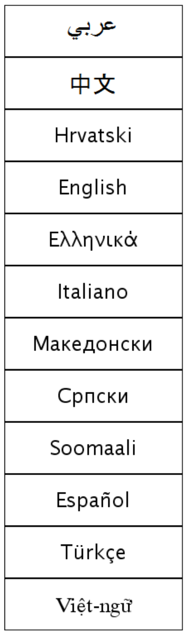
It is important to mention that non JavaScript and non Java solutions usually do not support the interaction with the eID (i.e., do not allow to authenticate or digitally sign votes with the eID). This is important to consider in case this is the required authentication and digital signature method for the voters.

If a pure JavaScript implementation is used (accepting the side effects of this approach) only JavaScript will be required. If server side voting client approach is accepted, the client machine requirements are reduced to just a HTML compliant browser.

These optional configurations can increase the number of supported platforms but introduce other side effects that must be carefully evaluated.

Elaboration of Requirement AU3

In the prototype, the language is set to Norwegian by default but it can be changed to English through configuration. For the final version, it would be possible to select multiple languages, for example via an icon, link, etc. The voting frontend for evoting provides a multiple language system. As an example, in a project that Scytl conducted for the Victorian Electoral Commission in Australia, the voting frontend was provided in twelve different languages:
Arabic, Cantonese/Mandarin, Croatian, English, Greek, Italian, Macedonian, Serbian, Somali, Spanish, Turkish and Vietnamese.
There are a number of characteristics that are important when providing a multi lingual user interface such as for example Unicode support, bidirectional awareness, fonts that are supported in one language but not in others, images may have different meanings in different languages, text sizes may be different in different languages, mirroring awareness for left-to-right and right-to-left display, capitalization (not all languages have the concept of lower- and uppercase), filtering out illegal character combinations, line and word breaks is handled differently in some languages, etc.



عربي
中文
Hrvatski
English
Ελληνικά
Italiano
Македонски
Српски
Soomaali
Español
Türkçe
Việt-ngữ

Keyboard variations and settings are handled by the Operating System. This is transparent to our solution. The provided solution uses UTF-8 encoding, which is a Unicode variable length character encoding. It can represent all characters of the Unicode standard, while it is backwards compatible as mentioned in the requirement. In UTF-8, each character is encoded in 1 to 4 octets, with the single octet encoding for US-ASCII characters.

The admin system supports multiple languages with the assumption that the supported languages are read from left to right.

The contractor will provide a suggestion of texts for the languages Norwegian (bokmål) and English. The customer will provide further language translations.

Standard software (i.e. iReport) will not support multiple languages, due to challenges when updating to new versions.

Elaboration of Requirement AU4

The solution has no restrictions on what types of documents that can be uploaded and presented for the user. We will ensure that when the user is given the opportunity to download printable/modifiable files, the user will be given the option to choose among various file types all presenting the same content.

Elaboration of Requirement AU5

As outlined in AU 1 and AU2, we are in full control of the front-end of both the e-voting solution and the administration module, and can therefore make sure that the pages are not too heavy to download.

It is important to notice that the provided solution can be designed to be used with low speed Internet connections. Please note that The Contractor has delivered Voting Clients with reduced functionality for low memory and reduced bandwidth devices (Java mobile phones), one had a size as small as 60 Kb.
Many electors have already successfully cast their votes through low speed and low quality Internet connections. Feedback from voters has always been very good. This includes voters in almost all African and Latin American countries for different binding public elections. In most cases the Internet connections available to many voters from these countries are low speed and low quality. Many voters still use 28.8 Kb and 56 Kb dialup modems to connect to the Internet. Another example is the first citizen e-consultation in Madrid in 2004. 52% of voters used a low speed and low quality Internet connection and the satisfaction rate regarding the system and access time was greater than 90%.

Elaboration of Requirement AU6
This requirement is fully supported. The electronic voting system does not store any information on the voter's voting device. The voter can therefore use any PC he/she wishes. The same is true for users of the electronic voting system administration, mixing and counting components.

Elaboration of Requirement AU7
As outlined in AU 1 and AU2, we are in full control of the front-end, and we are able design a solution that allows for various resolutions, this includes 800x600, 1024x768 and 1280x1024. Scaling between resolutions will be defined with the customer in the initiaton phase, to establish the best visual result for each resolution level.

Elaboration of Requirement AU8
The system will present the user with the option to change the font size. Major browsers (i.e. Internet Explorer, Firefox and Opera) support scaling of text through hot keys (CTRL+ and CTRL-). If it is desirable to scale this way, the hot keys can be "trapped" to provide zooming if the browser supports this functionality. The text and fields of regular user interface will have a high degree of contrast: Black text on white background. However we will also provide other color color/contrastoprions according to specification. The most likely candidate is to provide an inverse image with black background and white/yellow/light green text.

Elaboration of Requirement AU9
The voting system can be used with a mouse pointer and/or with the keyboard. The voting system also provides keyboard shortcuts. Keyboard only navigation and use was successfully tested and works without any problems. Use of mouse pointers and keyboard is consistent and standard throughout the application. Likewise, use of entry fields, radio buttons and drop-down lists is also consistent and standard throughout the application. Consistency and standard behavior and functionality of mouse pointers, key boards and user interface elements in the final solution will fully be designed complying with usability and accessibility standards and guidelines.

Elaboration of Requirement AU10
Sans serif fonts will exclusively be used in the solution. Style sheets are designed to use font-families, so that if a user does not have the required font installed, a similar one will automatically selected by the browser. The readability of the user interface will be tested on visually impaired users.

Elaboration of Requirement AU11
It is our understanding that we are requested to use the Common Authentication Infrastructure provided by DIFI for user authentication (i.e. logon). Thus, the actual logon page will be provided by that infrastructure. We will

receive authentication token from the CAI with information about the user's identity, the level of authentication as well as the preferred language.

Elaboration of Requirement AU12
From the user interaction point of view, the solution will not require any other GUI technologies than a web browser. However, the cryptographic operations required to protect the vote are implemented in a Java applet. These cryptographic operations do not require any interaction with the voter and therefore, no GUI is required for Java applet operations. The results of these cryptographic operations are returned to the browser that shows them as part of the HTML page (i.e., accessible by standard browser screen readers).

From the technical point of view, voters are required to have a browser and a Java virtual machine installed in their computers. However, voters with visual disabilities are not required to additionally install any Java accessible technology (i.e., Java Access Bridge) for voting.

Elaboration of Requirement AU13
The use of Java Script will be kept to a minimum and only used where it will bring considerable advantages to the majority of users and at the same time not create obstacles for users utilizing assistive technologies. If strictly necessary, a fall back solution without JavaScript is possible by using only server-side scripting and use of return codes. Elaboration of AU2 provides a suggestion of alternatives, with the goal of obtaining selecting a final solution during the design phase of the project.

Elaboration of Requirement AU14
The performance and scalability of the Election System as a whole is further explained in the elaboration of requirement P 1.1.1. For the presentation layer, this will be achieved in different ways. Calls to the service layer will be performed in a stateless manner. This makes it possible to load balance requests to different server (horizontal scalability). In addition, front-end caching in the web servers will provide low latency and high performance for static content. This will reduce load on the back-end systems, freeing up resources that can be used to provide dynamic content.

The System will be able to process all regular requests and provide feedback to the users within 2 seconds. A message will be provided while executing tasks that are not expected to be executed within this period. In this case all buttons are disabled. This gives the user a visual feedback that the system is processing the request; in addition it makes it impossible to execute the same operation twice.

Tasks that may have a longer response time than 10 seconds must be confirmed by the user before being executed. A dialog box with the expected response time will be provided, with options to proceed or cancel the operation.

5. Elaboration of Security Requirements

Elaboration of Requirement OS 0.4
The following security controls are implemented to prevent the counting of multiple votes per voter and contest, even if a failure occurs in the system:

- There is a central electoral roll that registers the voting status of the voter including the voting channel used to cast the vote (p-vote, e-vote controlled environment and e-vote uncontrolled environment) and, in case the vote has been cast electronically, the digital certificate used to sign the cast the vote.
- Electronic cast votes are digitally signed (after being encrypted) using a unique voter digital certificate. From remote uncontrolled environments, the voter eID will be used for digitally signing the vote. In controlled environments (i.e., polling stations) the voter could use the eID or a specific unique voter digital certificate valid only for that election. When using a special voter certificate, this will be based in a unique, non-transferrable and non-reusable private key per voter. The information of this special digital certificate assigned to the voter to cast his/her vote is stored in the voter entry of the electoral roll to allow later to validate the voter who cast that vote.
- The vote information digitally signed by the voter contains the identification of the contest the vote belongs to. The identification of the content is unique and follows the EML recommendations. This prevents using the same vote of the voter in a different contest.
- The e-voting counting stage starts with a cleansing process that selects the last final valid cast vote from voters that did not also cast any p-vote or e-vote in a controlled environment (as specified in requirement F 3.4.1). If a voter has a cast an e-vote cast from a controlled environment this vote will be selected only if the same voter did not cast a p-vote.
- Any information transferred between the different election modules that manage the vote collection and counting is digitally signed by the origin of the transfer. Therefore, these digital signatures can be used to verify the integrity of the information before processing it. This prevents the processing of any information that could be corrupted after a failure of the system.

In the specific case of e-counting (UC 3.4) the solution implements a universal verifiable re-encryption mixing process that provides cryptographic proofs to prevent the deletion or alteration of votes during the anonymization and decryption process (see Appendix 2A Amendment 1 for more details). More concretely:

- Prevents the elimination of votes when shuffled and re-encrypted by the Mixing process by means of using cryptographic integrity proofs of the input and output votes.
- Prevents the manipulation of pairs of votes by checking the previous proofs on random groups of the input and output votes. Any manipulation of a non-pair of votes is always detected by the previous mechanism.
- Prevents any manipulation of the proofs by means of ZKP of the proof contents.

In addition to the Mixing process, a ZKP of correct decryption of the vote contents or accumulated result is always generated. This allows for verification if the clear text vote corresponds to the cipher text used in the decryption process without disclosing the secret key used for its decryption. This prevents the manipulation of the vote by changing the decryption algorithm.

<u>Elaboration of Requirement OS 0.6</u>
This requirement is achieved by cryptographic means and authorization controls.

Cryptographic means are based on protecting the information by digitally signing it. Digital signature practices comprise:

- Digital signature of votes: prevents any manipulation of the votes once they have been cast. Votes are digitally sign by using a unique voter digital certificate. Therefore, any manipulation requires access to the voter private key.
- Digital signature of ballot boxes: Ballot boxes are digitally signed once the election is closed. This prevents any manipulation of its contents once the election is closed (e.g., deletion, addition or manipulation of votes in the ballot box). This process is also used to protect the ballot boxes produced

after eliminating the duplicate votes. The integrity during the election is managed through an immutable log approach.

- Immutable logs: Protects the integrity of the ballot box during the election. A hash of each stored vote is recorded in this immutable log and cryptographically chained with the previous one. The integrity of the contents of the ballot box can be checked at any time by verifying if the chain of hashes recorded in the immutable log corresponds to the votes stored in the ballot box.
- Digital signature of decrypt votes/results: The decrypted votes or results (when homomorphic tallying approach is used) obtained from the encrypted votes are digitally signed to prevent any modification after being decrypted.
- Universal verifiable mixing: The correct behavior of every node in the Mix-net used by the Mixing Process is universally verifiable. This prevents any manipulation of the votes during the shuffling and re-encryption process.

The solution also issues a voting receipt to voters, allowing to types of verification:

- Presence of her vote in the ballot box collected by the e-vote counting process. This process can be done with any vote cast by the voter).
- Presence of the voters vote in the final tally. This process only can be done with the last valid vote of the voter.

The first verification process can be done in public (i.e., publishing the list of voting receipts of the votes present in the ballot boxes downloaded from the vote collector servers). The second verification process should be done in an access restricted environment to prevent facilitating.

The voting receipts issued to the voters during the voting process and the published list of ballot box vote receipts are digitally signed to prevent any manipulation.

Authorization means are based on the role based mechanism. This security control mechanism is used to prevent any non-properly authenticated and authorized user from accessing the encrypted votes, decrypted votes and results (securable objects) information at any stage of the election. It is important to note that even the previous cryptographic measures are also efficient in case of internal attacks from any authorized user.

<u>Elaboration of Requirement OS 0.9</u>

The election system structure and functionality has been designed to be fully functional in case one of their components fails. Practices used are:

- In all systems, data transactions are designed to support rollback in case of failure to keep the information in a consistent status.
- All systems can be deployed in servers with redundant components.
- Critical network systems that require always staying online are functionally designed to support and implement clustering or load balancing practices (e.g., remote e-voting platform components).
- Critical networking systems that support temporary offline status are designed to support synchronization of information after reconnecting (e.g., electoral roll terminals at the polling stations can support temporary disconnections).
- Critical systems are designed to work on a two node high availability mode (e.g., multiple remote vote collector servers can manage voting transactions at the same time).
- Critical systems that require managing large amount of parallel transactions are designed to run in parallel over the same data repositories (e.g., the different steps of the same voting session could be managed by different vote collector servers).

- Databases of critical systems are designed to support clustering (e.g., electoral roll database, remote voting system database, etc.).

Elaboration of Requirement OS 0.12
This requirement is implemented using the following security controls:

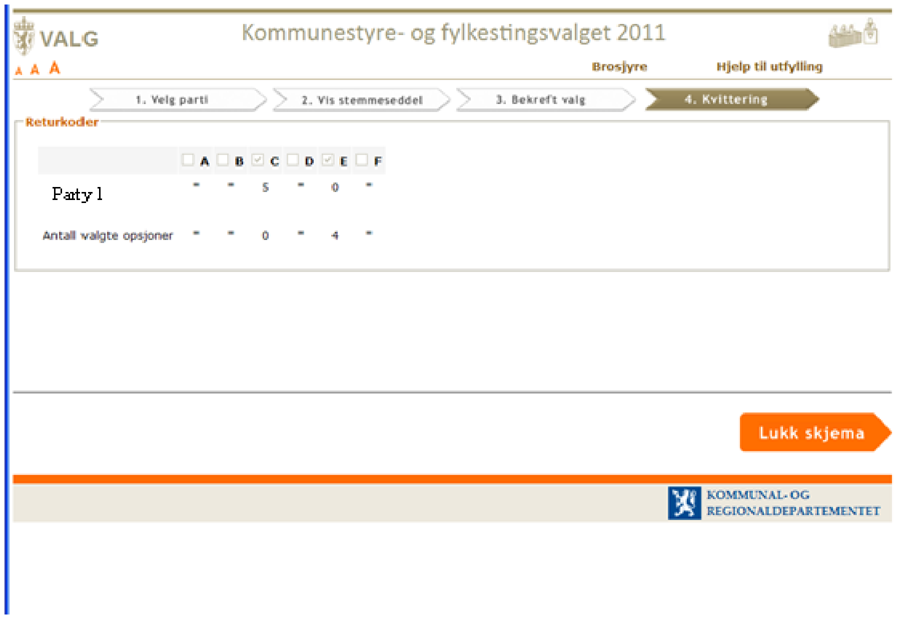
- The voter is provided with a voting receipt that unquestionably proofs to the voter that the vote has been successfully cast and furthermore can proof that the vote has reached the electoral authorities. The voting receipt is not linked to the contents of the vote. It is digitally signed by the electronic voting system and the signature is validated by the Voting Client on the voter's machine. At the end of the election, the receipts of the votes that reached the counting phase are published to allow voters to verify the presence of their votes in this process. In addition, the voting receipts of the votes counted (i.e., decrypted by the Mixing server) can be checked in a controlled environment to prevent coercion practices.
- The correct contents of the encrypted vote are verified before being accepted by the Vote Collector service. This verification consists on checking if the contents of the encrypted vote contain a valid option without disclosing this option (i.e., preserving voter´s privacy). This prevents voters from casting votes that could invalidate the process or put in question the accuracy of the counting process.
- The Mixing process is universally verifiable. Therefore, it is possible to check that the Mixing process did not manipulate the votes during the anonymization process (shuffling and re-encrypting votes).
- The vote decryption process generates a zero knowledge proof (ZKP) of decryption that ensures that the decrypted voting options are the ones contained in the encrypted vote.
- Additionally, the proposed solution can challenge the server using the return code to verify its authenticity based on a subset of codes only known by the voter and the server. This challenge, which can be validated by the voter, indicates that the vote was successfully received and processed by the authentic electronic voting system.

<u>Elaboration of Requirement OS 0.12B</u>

Figure - Return Codes shown to the voter

In addition to the voting receipt and the universal verifiable Mixing, the system can achieve these requirements by using return codes.

During the election configuration process, unique vote cards are created for each voter. These vote cards will contain a unique Ballot Identifier (BID) and one Return Code for each possible option (party and/or candidate). One voter card can only be used by one voter to verify the proper treatment of one or more votes cast by the same voter. At the same time, one voter can use different voter cards for verifying different cast votes. However, the same voter card cannot be used by different voters to cast their votes to prevent coercion attacks



Returkoder

	☐ A	☐ B	☑ C	☐ D	☑ E	☐ F
Party 1	-	-	5	-	0	-
Antall valgte opsjoner	-	-	0	-	4	-

(i.e., that a coercer give the same vote card to different voters to verify their cast votes).

Once the vote is cast and the voter requests verification, the verification process proceeds as follows:

- Voting client sends encrypted proofs of selected voting options to the voting platform.
- Voting platform verifies that these proofs are correlated to the cast vote to ensure that the return codes will belong to the same stored options. This is done using ZKP protocols to prevent the disclosure of the voting options.
- Return Codes are calculated by an independent component of the voting platform (validation service) using the encrypted proofs.
- Return codes are sent to the voter allowing him/her to check that it is the same ones as on the vote card.

The return codes are related to the selected options (e.g., party, candidates) and the number of selected options (e.g., 1, and 2). It is important to note that the code related to the number of selected options is generated by the voting platform without knowing how many options have been selected by the voter. The verification of the number of selected voting options prevents the inclusion by a rogue vote terminal of additional voting options without the voter knowledge.

The return codes can be sent to the voter through the same communication channel used to cast the votes or an independent one. In the former case, only a voter selected subset of two digits of the return code will be returned to prevent any eavesdropper learning the complete code (the voter can use a different subset in case he/she wishes to cast another vote). In case an independent channel is used, the solution can be integrated with SMS or email gateways for sending the return codes.

The combination of the proposed protocol using return codes, voting receipts and universal verifiable mixing provides and end-to-end proof that the cast vote is received, recorded and counted as the voter intended.

The cryptographic details of the protocol used to generate the return codes can be found in the description of the protocol implemented in the solution (see Appendix 2A Amendment 1 for more details).

<u>Elaboration of Requirement OS 0.13</u>
Voters are provided with a conformation screen that contains their selected voting options for confirmation before casting their vote. The review process is under the control of a trusted environment (digitally signed applet).

After casting their votes, voters are provided with a voting receipt that is digitally signed by the Vote Collection Server in case the vote is accepted. The voting receipt allows voters to verify that their votes have been delivered by the voting system to the electoral authorities. This voting receipt can be also used to verify if the vote reached the counting process.

<u>Elaboration of Requirement OS 0.14</u>
Votes are encrypted in the voting terminal using the Election public key. Therefore, the contained voting options can only be decrypted at the end of the Election by the Electoral Board members collectively. The same requirement is fulfilled by any verification process implemented by the voting protocols:

- Zero Knowledge Proofs (ZKP) of vote content correctness does not disclose the information stored in the encrypted vote.
- Voting receipts are not obtained from the clear text selected options and therefore, cannot be used to disclose the selected voting options.

<u>Elaboration of Requirement OS 0.15</u>
The solution provides the synchronization with a timeserver through the Network Time Protocol (NTP) and broad support for publicly available hardware based time server products

The solution provided by ErgoGroup/Scytl provides an outstanding accuracy of the time source to maintain time marks for audit trails and observations data, as well as for maintaining time limits for registration, nomination, voting and counting. To prevent issues related to local time at different time zones or daylight saving time changes, the time information is managed in Universal Time (UTC).

<u>Elaboration of Requirement OS 0.17</u>
The lists for production of ballot papers will be protected by using a encrypted and digitally signed PDF-format file. Nevertheless a production test of ballot papers is needed to verify that ballots fulfill the requirements for electronic counting of pvotes, e.g. adjustment marks and print quality. This verification will also include a content check of lists and candidates.

Elaboration of Requirement OS 0.18
The proposed solution allows the voter to abort the vote casting process without losing his/her right to vote at any later point.

The system does not store the vote, does not check off the electoral roll and does not update any right until the voter explicitly indicates that he/she is satisfied with the vote and wants to cast it. Only at that moment the system starts processing the cast vote and updating the database. That means that until that moment, the voter can at any time abort the voting process without losing his/her right to vote.

Storing cast votes and checking off votes in the electoral roll takes place inside an atomic transaction.

Elaboration of Requirement OS 1.2
The system uses unique identifiers for users. Voters are identified in the system using an internal unique identifier that is usually obtained from an existing one (for example the SSN). Otherwise, the internal identifiers are obtained from the combination of their personal information (name, surname, address, birth date, etc.) and election identification (using a hash function).

Election managers have also unique identifiers. Common access control policies are based on roles to prevent sharing accounts among different users.

Elaboration of Requirement OS 3.1
The system guarantees this requirement by means of:

- Encrypting votes at the voter terminal: voting options never leave the voting terminal unencrypted and can only be decrypted by the collaboration of the Electoral Board members.
- Using ZKP for verification of contents: verification of the correctness of the encrypted vote does not disclose the contents
- Using Schnorr signatures of the cast votes: this signature prevent that the contents of the vote encrypted by a voter could be disclosed in case another voter re-use the same encrypted vote
- Using a universal verifiable re-encryption mixing: breaks any correlation between the decrypted votes and the casting order
- Using homomorphic tallying of groups of votes: prevent disclosing the contents of an individual vote but the combination of multiple ones.

Elaboration of Requirement OS 3.3

The proposed Mixing process ensures that at no time any clear text vote can be correlated with the original encrypted vote of the voter or the casting order. The voting receipt use also prevents any correlation between the vote and the voter.

On the other hand, the system provided by ErgoGroup/Scytl allows defining the minimum number of votes that must exist in any electronic ballot box before it is being decrypted. If the minimum number of votes is not reached, then the votes of that ballot box are not opened to guarantee voter privacy.

In case that this situation is caused by a segmentation of the same election in regions, these votes can then be mixed with other votes to guarantee voter privacy. (e.g., in case that votes are segmented in countries for counting, and the votes of one country did not reach the valid limit, ballot boxes from different countries can be joined before decrypting the results).

Elaboration of Requirement OS 3.4
As mentioned in OS 3.3, the solution includes a universal verifiable mixing process and voting receipts that prevents any correlation between votes and voters. In addition, the solution provides the homomorphic tallying of groups of votes to prevent disclosing the individual information of a vote alone if required (it is important to note that this feature is not possible with traditional elections).

Elaboration of Requirement OS 3.5
The confirmation of successfully recording of a vote is based on the digitally signed voting receipt. This receipt cannot be used to disclose the voter intent.

Elaboration of Requirement OS 3.6
Any audit information of the cryptographic protocol that is required for end-to-end verification susceptible of being stored in the audit system, does not pose any security risk of the voter anonymity:

- Any vote information stored is based on the encrypted vote and therefore, it is not possible to discern the vote contents.
- Cryptographic proofs used to verify the correctness of the vote are based on ZKPs and therefore, do not allow to discern the vote information.
- Digital signatures of the votes are done over the encrypted vote and therefore, even though the owner of the vote could be disclosed, the vote contents remains anonymous.
- Voting receipts are not generated using the vote contents and therefore, cannot be used to disclose the selected voting options.
- The integrity and accuracy proofs of the mixing process are based on ZKPs generated over random sets of encrypted votes that change between mix-nodes.
- Mixing process is composed of four mix-nodes that permute the encrypted votes using different random numbers and therefore, the votes in the output of the mixing cannot be related to the ones in the input.

Furthermore, the audit process is universal and therefore, it is not required to use any private key for checking the accuracy of the counting process.

Elaboration of Requirement OS 4.1
The system ensures the secrecy of the vote at all stages of an election using an e-voting protocol based on advanced cryptographic algorithms that implements the following mechanisms:

- Encrypts votes at the voter terminal: voting options never leave the voting terminal unencrypted and only can be decrypted by the collaboration of the Electoral Board members.
- Uses ZKP for verification of contents: verification of the correctness of the encrypted vote does not disclose the contents
- Uses Schnorr signatures of the cast votes: this signature prevent that the contents of the vote encrypted by a voter could be disclosed in case another voter re-use the same encrypted vote
- Using a universal verifiable re-encryption mixing: breaks any correlation between the decrypted votes and the casting order generating completely different cipher text of the same vote votes but sorted in a different order.

- Uses homomorphic tallying of groups of votes: prevent disclosing the contents of an individual vote but the combination of multiple ones.
- Uses ZKP for universally verifying the accuracy of the Mixing process: this proof allows this verification without disclosing information that could allow link decrypted votes with original encrypted ones.
- Provides Homomorphic tallying of the votes: provides decrypted intermediate results of groups of votes instead of the individual contents.

Elaboration of Requirement OS 4.2

The solution incorporates the following security controls to handle the keys, passwords and other confidential (here and after Securable Objects) information:

- For Secure Objects that can be protected with standard access control measures:
  - Role based access controls: to restrict public access to certain confidential information (e.g., the number of votes cast by a voter)
  - Stored procedures at a database level: to prevent the direct execution of DB queries that could compromise the secrecy of confidential information (e.g., the e-vote results of a small district with a low participation rated).
- For securable objects that requires advanced access control measures, cryptographic based measures supported are:
  - Encryption using Hardware Security Modules: To physically protect the access to private keys during the encryption process (e.g., voter digital certificates)
  - Secure Multiparty Secret Sharing Schemes: For requiring the collaboration of several members for distributed decryption or digital signing of information (e.g., Electoral Board members).
  - Re-encryption of information using different keys: to prevent the disclosure of information to third parties (e.g., the re-encryption by the Vote Collector Server of the proofs of selected voting options coming from the voters before sending them to the Validation server)
  - Verification based on Zero Knowledge Proofs: to prevent requiring access to the private key used to encrypt the information for verifying the correctness of the information

Elaboration of Requirement OS 4.3

At the voter register level, privacy is achieved by means of combining cryptographic means and access control measures:

- Any information imported/exported between two different systems is encrypted and digitally signed. For example, the Electoral Roll information interchanged between the Election Configuration system and the e-counting system are encrypted and digitally signed files.
- Any online access to voter register information is encrypted and mutual authentication using SSL.
- Application level access to register information data is controlled by defined interfaces and/or db stored procedures.
- Any access to voter register information requires the authentication of the entity that requests this access.
- Access control to the information of voters from authenticated entities is managed by means of security policies based on roles.
- Access to critical information is additionally protected by means of encryption.

<u>Elaboration of Requirement OS 4.7</u>
The cryptographic asymmetric algorithms used are:

RSA:

- Key usages: digital signature, authentication and encryption (if information must be imported/exported between systems)
- By default 2048 bits but 1024 is also supported when digital certificates are not provided by the system (e.g., eID of voters)
- Digital signature algorithm by default based in RSA-PSS encryption scheme with SHA2 hash. PKCSv1.5 signature scheme with SHA1 is also supported for compatibility with other systems.

ElGamal:

- Key usage: encryption and re-encryption
- At least 2048 bits
- A generator g selected as the generator of the q-order subgroup Gq of Zp*

Symmetric algorithms used:

AES:

- Key usage: encrypt information transferred between (digital envelopes)
- Key length by default: 256

MAC functions used are:

MAC- SHA2:

- By default 256 keys but larger supported

<u>Elaboration of Requirement OS 4.8</u>
System components use SHA2 algorithms for MAC, immutable logs and digitally sign information. SHA1 is supported when interfacing with external systems that do not support SHA2 algorithms (e.g., digital signatures from eID if SHA1 is not supported by the PKI infrastructure).

<u>Elaboration of Requirement OS 4.9</u>
E-votes are secured stored and submitted using the following security controls:

- Votes are encrypted and digitally signed in the voter terminal and stored with the digital signature
- A digitally signed voting receipt is sent to the voter as a proof of vote storage. This receipt can be used to check if the vote was properly stored in the e-vote counting phase.
- The storage system is redundant and clustered to prevent losing any vote storage information in the case of component failure
- Periodical backups of the vote storage database are implemented by the system to allow the recovery of the votes in the case of a complete disaster
- An immutable logs registers the integrity of the vote storage to detect any manipulation of the votes

The e-voting solution has been designed to not pose any limitation in terms of system redundancy at infrastructure level. Therefore, e-vote storage redundancy measures as the following ones are supported:

- Server with redundant components running in fault tolerant mode (i.e., the failure of one redundant component does not imply the stop of the service or loss of information) and hot-swappable (faulted components can be changed without stopping the system).
- All file systems are configured in transaction mode.
- RAID disk storage: the storage system used for ballot box storage and backup is configured using a RAID 1 with hot spare disc approach.
- Redundant disk access paths: controllers and access paths used to access the disk enclosure are redundant and configured in fault-tolerant mode.
- Disk backups: E-vote ballot box and its integrity logs are periodically (e.g., 5 minutes) backed up to disks.
- Tape backups: Disk backups are periodically backed up to tapes or other removable media (e.g., every 30 min.).
- Redundant cluster database system: Database system is implemented using an active-active cluster approach. Any changes in one cluster database node are automatically synchronized to the other node.

At the application level, the e-vote is stored in the database in a transactional and synchronous way. This guarantees that the database transaction is not considered as finished until each database node finishes its transaction. The application also provides the parallel storage of the e-vote backups in a secondary store disks (in addition to the database backups implemented).

The receipt is sent to the voter when the vote is successfully stored in both nodes of the cluster and the ballot box integrity log updated. The application can also wait with storing a backup of the vote in a secondary disk backup media (this is configurable).

Since the solution does not pose any limitation in terms of system redundancy and replication, the RPO that could be achieved by the solution is 0. However, the real RPO provided will depend on the final infrastructure deployed and SLA provided by the datacenter.

Supported practices to reach an RPO practically equal zero are:

- Setup to backup the e-votes in a secondary media as soon as the votes are stored in the database.
- Use RAID 1 or similar approach on all the disks and hot-spares.
- Setup a database cluster with multiple nodes: the longer number of nodes the better RPO offered.
- Setup up redundant storage cabinets: the longer number of replicated cabinets the better RPO offered.
- Setup up as many hot-spares disks as possible
- Setup up as many secondary backups disk storage as possible
- Use the faster removable backup systems for allowing smaller backup periods.

The RTO will depend on the recovery speed of the removable backup media (considering that all the redundant storage measures, including disk backups, fails). The final RTO of the solution will depend also on the final infrastructure used in the deployment and SLA provided by the datacenter.

To reduce the RTO time the following practices are:

- Disk image of the servers
- Spare units of the servers
- Disk backups of the information
- Replicated arrays systems
- Replicated Datacenter system (Disaster Recovery Site)

Since the solution provides database clustering, component fault tolerance, RAID configurations and disk hot spares, re-establish redundant e-vote storage after a complete system failure of the primary e-vote store during an election is in most of the cases the recovery is immediate (assuming that secondary storage is always synchronized with primary). For instance, the recovery of a lost disk is immediately started over a hot spare as soon as detected.

In most of the cases no (e.g., a disk failure), e-voting service will be available while re-establishing redundant e-vote storage, since the recovery operations can be performed while the redundant e-vote storage is connected (e.g., hot spare or database cluster synchronization). In case the information must be recovered from a backup media, the system can register new votes while recovering if the Electoral Roll information is not affected by the recovery process.

<u>Elaboration of Requirement OS 4.11</u>
The system provides the following security controls to protect private or secret keys according their importance:

- Software key containers: Password based encryption containers to store private keys (e.g., PKCS#12).
- Hardware Security Modules (HSM): Hardware based FIPS 120-2 compliant cryptographic key storage (e.g., smartcards)
- Multiparty computation secret sharing schemes: Cryptographic schemes that allow splitting a private key in shares (without generating this private key) that can be distributed among different users. A predefined threshold of users should be required to perform distributed operations equivalent to using the private key or the reconstruction of that key.
- Physical access control based on secret sharing: HSM with support of Access Control list for accessing internal stored private keys.

More specifically, the following key storages are supported by the system

1. - Voter keys:
   - E-ID: Smartcards provided by the government. Private keys are stored in the smartcard.
   - Key roaming: Software PKCS#12 files encrypted using a random key. This key is encrypted at the same time with the public key of the election officials that manage the Electoral Roll in the controlled environments. These digital certificates are only used in controlled environments and stored in smartcards after their decryption by the election official (using his/her smartcard).

2.- Service keys
   - HSM: The service private keys can be generated and stored inside an HSM.
   - Software key containers: The system also provides the use of software key containers (PKCS#12 or Java key storage) if required by the customer.

3.- Election manager keys
   - Smartcard: Election manager keys or key shares (when separation of duties is implemented) are stored in cryptographic PIN protected smartcards.
   - Software key containers: The system also provides the use of software container storage if required: Windows key container, web browser (e.g., Firefox container), PKCS#12 container, etc.

Supported HSM and smartcards can be FIPS 140-2 compliant. In these cases, the key can be generated and stored in the HSM/smartcard. Therefore, it never leaves the internal hardware storage. The system provides nCipher and SafeNet HSMs.

The system includes a key management module that implements the key generation, protection and certification:

- Generation: keys can be generated in three ways:
  - HSM/Smartcard: in this case, the keys are generated inside the HSM/smartcard and the public key provided to the management module for its certification (using a CSR approach).
  - Distributed: in the case of Separation of Duties, the key management module allows the distributed generation of the shares without requiring the generation of the private key.
  - Software: the module also allows the generation and storage of private keys inside key software containers. In this case, the process is executed in an isolated environment using a cryptographically secure random number for the key pair generation and the key container password.
- Protection:
  - HSM/Smartcards: requires the introduction of a PIN or password to access the device.
  - Distributed: requires the introduction of a PIN in the smartcard where the share is stored.
  - Software: generates a random password (using the cryptographically secure PRNG) to encrypt the PKCS#12 file
- Certification:
  - Keys needs to be certified by an election trusted authority or committee. Keys must belong to any of the PKI trusted by the election.
  - Key usage: the usage of the keys should be restricted (e.g., encryption or digital signature or authentication)
  - Expiration: the validity of the key must be limited (e.g., to the election period)
  - Election use: the re-usability of the key for different elections can be prevented.
- Life-cycle:
  - Revocation: the key management module also manages any key incident that could require the revocation of a key.

Section 6.1 (Key Management) of “SSA-U Appendix 2A-Amendment 1 Voting Protocol” provide more details of the management of the keys used by the cryptographic protocol.

<u>Elaboration of Requirement OS 5.1</u>
The system provides means to allow voters to verify that they are connected, including:

- Verification of the SSL connection server: voters are able to check the authenticity of the voting front-end by verifying the digital certificate used by the server to establish the SSL connection (usually done by the web browser by default).
- Verification of the digital signature of the applet: voters can check the authenticity of the applet that executes the cryptographic protocol by checking its digital signature (usually done by the Java Virtual Machine by default)
- Verification of the digital signature of the voting receipt: in addition to the receipt identifier, the voter receives the value of the vote collector service digital signature of this receipt identifier. The voter can check this digital signature to verify if it has been issued by the official election system
- Server return codes: The system incorporates a unique server return code that is provided to the voter with the voting card. If the voter is connected to the official election system, the proper code should be returned to the voter by the system after casting her vote.
- Voting option return codes: In case end-to-end verification is used, the return codes of the voting options can be used by the voter to check if the voter has cast her vote in the proper election system.

Instructions on how to verify the connection are clearly provided to voter.

Elaboration of Requirement OS 5.4
Online systems use mutual authenticated connections to interchange the information. For instance, in the case of voters, server is authenticated using the SSL digital certificates and voters are authenticated using their digital certificates.

Offline connections (e.g., air-gapped systems) are validated by checking the digital signature of the exchanged information. For instance, digital signature of the election configuration information coming from the air-gapped election configuration component is verified by the voting system before accepting it. Also p-votes scanning results are handled this way. On the other hand, digital signatures of the ballot boxes coming from the voting services are verified by the air-gapped tallying system before processing it.

Elaboration of Requirement OS 6.1
The source code of the system is provided for public review. Instructions on how to compile it are also provided. The voting system components are compiled by independent auditors. The compiled binaries are digitally signed by the same auditors. In the case of the voting client and other Java based components, a jarsigner tool is used to digitally sign the byte code. This allows java virtual machine to automatically verify the digital signature of the Java applications before being executed.

More specifically, the security of the server side platform is fulfilled by following a three layer trusted chain. Following a bottom-up approach, the different layers are:

- Digital signature of the application binaries and configuration files. Protection of the election specific application binaries and configuration files by means of digital certificates (digitally signed applications)
- Operating environment logical sealing. Protection of the application server and election application execution environment by means of logical integrity tools (tripwire)
- Operating environment hardware sealing. Protection of the logical integrity tools by means of hardware integrity sealing mechanism (TPM)

1- Digital signature of the application binaries and configuration

Election system binaries are digitally signed java applications and application servers will validate the digital signature before executing them. Application servers will be configured to allow only the execution of the applications digitally signed by the valid digital certificates. To this end, all the digital certificates of the trusted key store of the application server are eliminated but the one of the trusted authority that signs the applications. These java applications are compiled and digitally signed by the trusted authority using the source code files audited by independent experts (and make them public if required).

2- Application execution environment logical sealing

To prevent the manipulation of the application server binaries or any other system components required to execute and interact with the election application, the integrity of these application servers and system components are logical sealed using tools like tripwire. Tripwire stores a database of the fingerprints of all the files stored in any directory of the system. This tool will be initialized (baselining) and configured to secure store and periodically check the integrity of the application server and system component binaries and configuration files. Reports of the periodical checks are sent to the audit system for monitoring purposes.

3- Operating environment hardware sealing

To prevent any manipulation of the tripwire binary and related files, a hardware TPM chip is used to generate and store the fingerprint of the tripwire binary files, configuration files and dynamic libraries executed by the kernel during runtime. The values of the fingerprint tripwire related files can be checked at any time by querying the TPM chip. This query (known as attestation) requires sending a random nonce to prevent reply attacks. The fingerprint list is returned with the nonce, all of them digitally signed by the TPM.

TPM chip is configured to provide integrity measures of the BIOS, boot loader (TrustedGRUB) and kernel (including the kernel executed files such as tripwire).

An additional measure used to prevent the execution of code that is not approved is the use of read-only media to boot the system (e.g., CD-ROM). This approach is used in environments where the operating environment do not requires read access to disc. An example is the voting terminals deployed in controlled environments.

<u>Elaboration of Requirement OS 6.2</u>
Regarding the system configuration, the election configuration information is digitally signed by a committee of Election Authorities after being validated. The digital signature process is based on a Secure Multi-party computation process that requires the collaboration of at least a pre-defined set of members for signing the information in a distributed way. The e-voting platform system component verifies the digital signature of the election configuration before accepting it. If the validation fails, the system component does not start.

The digital signature validation process done by the e-voting platform system component is an automated procedure. Also the verification of the digital signature of the same e-voting platform system component is automated. In this later case, the validation is done by the application server that executes this component. The application server is configured to automatically check the digital signature of the e-voting platform components before executing them. This check is equivalent as the digitally signature check of the java applets, but in this case the application server does not show any screen asking for manual conformation of the execution: if the signature is correct and issued by a trusted digital certificate is executed, otherwise an error is reported and the execution aborted. As part of the hardening process, all the default trusted authorities of the application server key store are eliminated. Only the digital certificate of the election trusted authority is kept to allow digital signature validation.

The voting system components are compiled by independent auditors. The compiled binaries are digitally signed by the same auditors. Checking the digital signature of the voting platform software it is possible to discern if only the approved and certified software is installed and running in the system.

Baselining is implemented by storing the fingerprints values of the files (binary and configuration files) managed at each of the different levels of the chain of trust in a baseline database. Chain of trust is composed by:

- Digital signature of the application binaries and configuration files. Protection of the election specific application binaries and configuration files by means of digital certificates (digitally signed applications)

- Operating environment logical sealing. Protection of the application server and election application execution environment by means of logical integrity tools (tripwire)
- Operating environment hardware sealing. Protection of the logical integrity tools by means of hardware integrity sealing mechanism (TPM)

This baseline database can be a text file, csv file or database file. This baselining is executed before the election starts, when the final version of the election system is ready. Baselining fingerprints will be used during the election to verify the integrity of the election platform. After baselining, the voting system is considered sealed and any changes would be considered as possible attacks.

Information included in the baselining consists of:

1- Digital signature of the application binaries and configuration

The digital signature of the voting system executable components (Java signed) and configuration files.

The digital certificate of the trusted authority that digitally signed the application files is also stored in this baseline database.

2- Application execution environment fingerprints

The initialization of tripwire generates a fingerprint database of the files monitored by this tool (specified in the tripwire configuration file). This includes at least:

- Election application files (even though these files are validated by the application server, tripwire also keeps a fingerprint of them to implement periodical checks).
- Application server files (binaries and configuration files of the application server that runs the election application)
- Web server files (binaries and configuration files of the web server, if any, used to execute the application front-ends)
- Database files (binaries and configuration files of the database server)
- Kernel files (binaries and configuration files of the kernel, system services and system libraries)

The tripwire database generated after its initialization can be stored as the baseline of the system configured in a server. Tripwire also allows export of the database information in other formats (e.g., text file).

3- Server BIOS, boot loader, kernel and kernel executed files fingerprints

Each time the server is booted, the TPM stores in its own registers the fingerprints of the BIOS, boot loader and kernel executed. This information can be obtained using the attestation process and stored in a database or file as a baseline for future queries. The verification of the specific files executed by the kernel during runtime (e.g., tripwire) is also obtained and baselined. In this case, if any file is modified at any moment during the execution process, the TPM automatically updates the fingerprint value of this file. Therefore, in the next attestation query this fact is detected. The TPM kernel extension also keeps a history of the fingerprinted files and this history is returned when the TPM is attested. Therefore, any fingerprint changes of a file during its runtime use are recorded and these changes can be detected even if the file is later recovered to its original state.

4.- CD-ROM media fingerprints

In case Live-CD is used to boot a system (e.g., voting terminals at controlled environments), the fingerprint of the whole CD-ROM (e.g., ISO file) is also baselined to allow future integrity checks.

Elaboration of Requirement OS 7.1

The system has functionality that grants a role access to securable objects. Securable objects are organized in a hierarchical way. Thus, access may be granted at any level in the hierarchy. When the function "grant access to securable object" is activated, a view of the hierarchical structure is shown and a securable object may be dragged and dropped onto the role.

Elaboration of Requirement OS 7.2

The system does not implement any anonymous access mode. Users have a unique identifier that univocally identifies them to the system. Roles also have unique role identifiers. The system requires always first to identify the person by the authentication method required by the service (e.g., critical services require digital certificates). Once properly identified access to the service and the relevant securable object is granted based on the role assigned to the user identity or the user identity itself.

Elaboration of Requirement OS 7.3

The system comes with a set of predefined roles. These are:

- Competent electoral authorities, i.e. the electoral board
- Technical operators, supporting the technical system
- Auditor, responsible for auditing the complete election process
- Observer, external observers of the election
- Candidate, candidates for the election
- Voter, the voters themselves
- Independent security body, responsible for verifying the security set up of the system

In addition new roles may be defined based on existing ones or created with a completely new template. The task of creating a new role (in addition to the predefined ones), including assigning the role-owner, must be done by a signed request from the "Competent electoral authorities" and actually implemented by the "Technical Operator". Thus, two roles/persons are always involved in creating a new role and assigning the role-owner.

Each role has some general attributes that are defined when creating the role:

- The role name and unique ID
- The owner(s) of the role
- The authentication method (i.e. the minimum authentication level required) for members of the role.
- The modification authorization procedure.
    - If the role-owner may add other users to the role on his own, or if collaboration with a System Operator is needed.
    - Which other roles must be involved when adding or removing access to the securable objects. (The role owner can never add securable objects on his own).

The following operations may be performed on a role:

- All management functions like suspend, delete, update and so-on.
- Add unique users to the role
- Add securable objects that the role may operate on
- Select one of the predefined permission profiles (i.e. permission levels) that this role has on the securable object.

All of the predefined roles in the system, except for the voter role, are mutually exclusive.

Elaboration of Requirement OS 7.6

The security measures implemented for the encryption and decryption of votes do not allow the correlation of a clear text vote with the voter. However, there are special cases (as in traditional elections) in which a low participation rate could compromise the privacy of voters. To prevent this situation, the voting system provides the configuration of rules that prevents the decryption of votes in case their numbers do not reach a given threshold. Another option supported is the combination of votes that do not fulfill the rules with others to obtain a consolidated result.

Elaboration of Requirement OS 7.7

The proposed solution implements Secure Multiparty key Computation (SMPC) schemes to split private keys in shares that are distributed among a set of members. The SMPC schemes implemented allow defining the threshold of members required to implement distributed decryption or signature of information using their shares. These distributed operations will be equivalent as the decryption or digital signature using the private key. The scheme also allows the reconstruction of the private key from a threshold of the share members if required. The solution implements SMPC schemes for RSA and ElGamal algorithms.

To protect the Election private key, the solution uses an ElGamal SMPC for the distributed generation of the Election private key. The shares of the private key are distributed among members of the Electoral Board that participate in this generation. This process is done without requiring the creation of the private key and therefore, the private key never exist during the Election only its shares. The share generation is implemented as part of the Election configuration process and shares are stored on PIN protected smartcards own by each Electoral Board member.

To decrypt the votes, a pre-defined threshold of the Electoral Board members needs to contribute with their shares to decrypt the votes. The number of members of the Electoral Board and their threshold is configurable and does not allow defining a threshold larger than the number of members. The supported number of members is larger than 5 (we used in some elections thresholds of 6/10).

Elaboration of Requirement OS 7.8

The system provides the storage of any private or secret key as well as shares in FIPS 140-2 level 4 compliant HSMs.

Elaboration of Requirement OS 7.9
The system implements secure multiparty generation of ElGamal keys using the improved Mauland, Reistad & Mjølsnes referenced proposal of the Pedersen's Distributed Key Generation protocol. It also provides the "Efficient generation of shared RSA keys" proposal on Boneh & Franklin using the Mauland, Reistad & Mjølsnes improvements.

Elaboration of Requirement OS 7.11
The default authentication method for remote connection is the digital certificate. Therefore, having access to authentication data (i.e., digital certificates) does not allow any unauthorized entity to successfully impersonate any user or voter: the private key is still under the control of the user or voter.

For any other authentication mechanism not based on strong authentication, such as local authentication using username and password, the password is stored in a double hash mechanism to prevent being guessed if the password database is compromised.

From the access control point of view, access to the systems that contain authentication is managed by roles.

Elaboration of Requirement OS 7.14
The system provides separation of duties within the group of election officers by means of the authentication. Authentication of members is based on a unique identifier and common or individual duties can be assigned by mean of assigning roles to the identity of the member.

The separation of duties is based on requiring the collaboration of more than one person to access to a private key. This private key the can be used to decrypt information that was previously encrypted with the corresponding public key to enforce separation of duties. Separation of duties is also used for requiring the collaboration of several members to digitally sign information (e.g., election configuration information). The mechanism implemented, allows definition of a threshold of a group of members for making the encryption/digital signature operation. Therefore, in case one of the members is missing the encryption/digital signature could still be done (i.e., permanently losing one member does not prevent to decrypt/digitally sign information forever).

The proposed solution provides two different separations of duty approaches based on threshold schemes:

- HSM based: In such approach, the HSM is setup in a way that a subset of members must introduce an authentication smartcard to allow the system to use the private key stored in the HSM. Therefore, the HSM internally keeps an Access Control List (ACL) with the identity of the members of the group and the threshold of members that must be authenticated.

Cryptographic based (i.e., secret sharing): in this approach, the private key is not stored in any place. Secure multi-party computation techniques are used to create a set of shares that can be used in combination to execute cryptographic operations that combined have the same result as using the private key. Section 6.1.1. (Electoral Board keys) of "SSA-U Appendix 2A-Amendment 1 Voting Protocol" provides more details about the secure multi-party computation protocols implemented for separation of duties.

Elaboration of Requirement OS 7.17
The system requires strong authentication based on digital certificates for administrator, operator and auditor access. Strong authentication techniques are based on challenge response method for authenticating a user. Therefore, these methods prevent the implementation of reply attacks (a different challenge is used each time

the user is authenticated). All the authentication processes implemented in the solution are based on this approach.

Proposal also provides two-factor authentication. Two-factor authentication is based on requiring the user to use at least two of three authentication components:

- Something that he/she knows (e.g., a password)
- Something that he she has (e.g., a private key)
- Something that he/she is. (e.g., biometric information)

Digital certificates when protected by a PIN or password can achieve the two-factor authentication. All the digital certificates used by the solution are protected by a PIN or password.

<u>Elaboration of Requirement OS 8.1</u>

Together with an adequate IT infrastructure, security policy and backup policy, the election system makes it impossible for e-votes to be lost under any circumstances:

- E-votes that are successfully cast and have been acknowledged are stored in a relational database. Storing cast votes and checking off votes in the electoral roll take place inside an atomic transaction. The acknowledgement is only provided to voters once the transaction is successful.

- Support of redundant and fault tolerant IT infrastructure prevents that any data that is stored in the relational database from being lost. An adequate backup strategy assures that data is available even if the redundant and fault tolerant IT infrastructure would fail.

- Data and log files can be written to WORM devices that ensure that the data cannot be overwritten.

In addition of ensuring that E-votes are not lost under any circumstance, the solution provided by ErgoGroup/Scytl furthermore provides auditors, election officials and voters with the ability to actually verify this:

- The main actions performed by the provided electronic voting solution are recorded and protected cryptographically in immutable logs. These logs, which cannot be tampered with without detection, do not break vote secrecy and voter privacy. The logs effectively allow auditors and election officials to ensure that no E-votes are lost.

- The voting receipts allow voters to self verify the correct treatment of their votes and significantly enhance the transparency of the complete electoral process. If the voting receipts are published, the voters can verify that their E-votes were not lost.

The solution has been also designed to support its implementation in fault tolerant environments:

- Systems with redundant components: systems (e.g., CPUs, disks, network cards…) that prevent the failure of the service in case of failure of one of the components.

- Systems with hot spare components: Systems with components (e.g., disks) that are substituted automatically by another component in case of failure, without requiring the intervention of administrators.
- Systems with Hot swap components: Systems that allow the substitution of failed components requiring to shutdown the system or services
- Clustered systems: Systems that prevents the loss of the complete service in case one of the cluster node fails (e.g., motherboard failure)

The solution architecture also includes fault tolerant practices that reduce the risks of data or service failure:

- Provides database clustering
- Includes backup practices
- Provides load balancing (to detect any system component failure and redirect automatically the connections to the other).
- Includes system/network status monitors based on service checks (allows the remote monitoring of the complete voting application).

The complete voting platform could be replicated in alternative sites (e.g., other ISP facilities). Both platforms can be synchronized, allowing resuming the voting process with the same recorded information in backup site with the minimum downtime available.

Relevant information is continuously replicated in backup media.

The voting solution has been designed for allowing the replication of services and facilitates the scalability of the system without requiring to shutdown the services.

Elaboration on the atomicity of voting transactions

The status of the voter in the electoral roll will be marked by the voting system transitionally and only committed when the voter finalizes the voting process and the vote is stored. The transaction covers both activities. If the voter does not cast a vote, then the update in the electoral roll does not take place.

When the voter goes to a polling place (controlled environment) and is authenticated by the poll worker and his/her voting status checked in the central Electoral Roll. If he/she is allowed to vote and has no eID, poll worker assigns him/her with a digital certificate that gets stored in the Election Electoral Roll as “assigned” to a voter. It is important to note that this assignment does not updates the voting status of the voter, only the credential used for voting. This certificate is then stored in a smartcard to allow the voter to vote in the voting terminal. If the voter has an eID, poll worker only checks if he/she is allowed to vote before allowing him/her to access to the voting terminals.

The voter votes in the terminal through the same e-voting platform as the remote voters. Once he casts his vote, the atomic transaction implemented by the Vote Collector Server covering both the update of the electoral roll and the casting of the vote will ensure that there is never a discrepancy between the number of voters in the electoral roll having cast a vote, and the number of votes in the voting servers.

When the voter returns the smartcard to the poll worker, the later can check if the voting status of the voter in the Electoral Roll has changed. If not, it means that the voter has made an error and the poll worker can allow him/her to try again. Voting status is not updated manually by the poll worker but the Vote Collector server.

More concretely, to ensure an atomic transaction for storing the vote and checking it off in the electoral roll, a two phase commit protocol is used. With this approach, the vote collection service, after verifying the signature of the vote and voter eligibility in the Electoral Roll, starts two operations within a single transaction:

- On the electoral roll service: to update the voting status of the voter.
- On the same vote collector service: to store the vote in the ballot box database.

The vote collector service requests a commit from the electoral roll service and from itself. If both services commit successfully, the transaction is finished and the voter can be informed that the vote was securely cast.

If one of the services cannot commit successfully, the transaction on the other service will be rolled back. In case the other service has:

- Not committed yet, this transaction is rolled back.
- Already committed, the transaction is rolled back using the undo log.

After all transactions are rolled back, the voter will be informed that the vote could not be cast.

The vote collector service acts as well as the controller of the two-phase commit (i.e., the vote collector has the role of the controller and at the same time needs to execute the transaction in the Ballot Box database).

The controller can handle several transactions simultaneously, and obviously, independent from each other. This is necessary in case of a clustered database, for example, where all the nodes need to be updated at the same time

Furthermore, the voting user interface will be designed in such a manner that the voters are guided through the complete voting process, ensuring that the voters clearly understand at which point the voting process has finished. This assures that a voter will remove his/her smartcard at the correct moment, once the vote has been cast, and not at an earlier stage. Continuous usability testing, built in the software development lifecycle, and the feedback obtained and implemented from these usability tests guarantee a self explaining and easy to understand user interface, ensuring that voters can fill out their ballot sheet without any issues.

ErgoGroup/Scytl also recommends the usage of smartcard readers that swallow the smartcard, making it impossible to remove it at any stage during the voting process. The system will only return the smartcard when the voting process has finished (e.g the voter has cast a vote) or aborted.

<u>Elaboration of Requirement OS 8.4</u>
Voters digitally sign their votes which makes it impossible to alter them during transfer in the network or at any other moment.

If during transfer in the network, the voter's vote is deleted, this is immediately detected by the voter because he/she does not receive the acknowledgement from the electronic voting system that his/her vote was successfully cast.

All voting actions are recorded in immutable logs, without breaking voter privacy or vote secrecy. Any deletion of vote records will be detected. It is also impossible to add votes because that would require access to the voter's certificate to digitally sign the votes. Vote injection is also impossible because the voter needs to sign it and because the system validates that there is only a single vote during the counting process.

Elaboration of Requirement OS 8.5

The votes are encrypted with the election public key (2048 bits) to maintain their confidentiality. Voters digitally sign their votes to guarantee their integrity and authenticity. Votes are very effectively sealed until the counting process. Only a threshold of qualified Electoral Board members is able to construct the election private key to decrypt the votes. A cryptographic Mixing process furthermore maintains the confidentiality of the votes.

The solution provides a redundant and fault tolerant IT infrastructure for secure storing of the submitted votes. Access to the IT infrastructure should be adequately protected by an effective security policy. This security policy should also physically protect any offline transfers. An adequate backup policy should be in place. Data and log files can be written to WORM devices.

Elaboration of Requirement OS 8.6

The integrity of the data communicated in, to or from the Election System is maintained in the following ways:

- In air-gapped data transfers, the information is digitally signed before being exported to removable media. This is for example the case of the Election configuration information.
- In online transfers, the connection established is mutual authenticated using digital certificates (SSL). As an additional integrity level, the information can be also digitally signed before being transferred.

Elaboration of Requirement OS 8.7

Voting application is a simple piece of code that can be easily audited and digitally signed by the electoral authorities. Voters can check the authenticity of the digital signature on the application before casting their votes. Additionally, the application generates a voting receipt that allows each individual voter to verify the correct treatment of his/her vote.

Voting application has been designed to be protected against external threats in the voting terminal:

- Voting client source code audit
- Digitally signed voting client application
- Implementation of the validation process in the digitally signed applet
- Require the use of voter digital certificates to cast a vote
- Server return code
- Voting receipt
- Client side code obfuscation

The system also provides the use of Return Codes for verifying the correct recording of the votes by voters. Once the vote is cast, the Return Codes verification process proceeds as follows:

- Voting client sends encrypted proofs of selected voting options to the voting platform.
- Voting platform verifies that these proofs are correlated to the cast vote to ensure that the return codes will belong to the same stored options. This is done using ZKP protocols to prevent the disclosure of the voting options.
- Return Codes are calculated by an independent component of the voting platform (validation service) using the encrypted proofs.
- Return codes are sent to the voter allowing him/her to check that it is the same ones as on the vote card.

The return codes are related to the selected options (e.g., party, candidates) and the number of selected options (e.g., 1, 2). It is important to note that the code related to the number of selected options is generated by the voting platform without knowing how many options have been selected by the voter. The verification of the number of selected voting options prevents the inclusion by a rogue vote terminal of additional voting options without the voter knowledge.

The return codes can be sent to the voter through the same communication channel used to cast the votes or an independent one. In the former case, only a voter selected subset of two digits of the return code will be returned to prevent any eavesdropper learning the complete code (the voter can use a different subset in case he/she wishes to cast another vote). In case an independent channel is used, the solution can be integrated with SMS or email gateways for sending the return codes. Below different approaches are discussed. The detailed design and requirement will finalize the actual solution selected. The pricing covers the selected option(s).

The proposed solution relays on voter verification of the correct recording of their voting options, to detect any Trojan attacks at voting client. The e-voting solution implements the vote casting and vote verification process as two different steps. These steps can be configured in two ways:

- Cast the vote first and then verify its proper recording: This is the configuration shown in the prototype and was designed to make the process more usable to the voter when multiple vote casting is allowed. The drawback of this configuration is that making the verification process optional will limit the number of voters that will verify their votes and potentially makes the success rate of a Trojan higher.
- Verify the vote first and then cast the vote: This is the configuration used when voters are only allowed to cast one vote per election. This approach implements the voter verification process before casting the vote (i.e., return codes to voter are shown always to the voter before casting). The main issue of this configuration is that it makes the voting process more complex to voters since they need to deal with the return codes before casting the vote. On the other hand, even though voters always pass through the verification process, it is important to note that this configuration does not ensure that the voter will really verify the vote, since he/she can still opt for casting the vote without checking the displayed codes (i.e., the verification process is still optional for the voter). Therefore, Trojan attacks could still successful change the voter intent if the voter does not really verify the vote. However, the fact that voters are always presented with the voting options increases the chances that they verify the vote.

Therefore, it is clear that if the voter does not really verify the vote (even though the verification process is implemented before casting the vote) a Trojan attack would not necessarily be detected. This is the case in the solution proposal but also in any other proposal, since it is not possible to ensure that the voting options really express the voter intent without the collaboration of the voter who cast them.

The only way to reduce the impact of Trojan attacks is to promote voters to participate in the verification process. One option is to use the ‘verify first and cast later approach’. To improve the usability of the process, the Vote Cards can be bound to voters and therefore, eliminate the requirement of introducing the Vote Card identifier/Sikkerhetskode. A unique identifier of the voter, such as the SSN or a key encrypted with the voter public key if supported, can be used instead. In this case, it is important to mention than the flexibility of generic Vote Cards is lost, since it is not possible for voters to randomly use a vote card. This could cause logistic problems in case voters lose their cards and require a copy.

To summarize, the only way to reduce the impact of Trojan attacks at the voting terminal is to promote voters to verify their votes before or after casting them. ErgoGroup/Scytl’s proposed solution is flexible from the customization point of view (without requiring changes to the e-voting protocol) and can be setup during the initial requirements definition of the project to better fit the security and usability requirements demanded by KRD. Current proposal is just one possible setup but other ones are supported without compromising the security of the underlying cryptographic protocol.

<u>Elaboration of Requirement OS 8.9</u>
The following measures allow detecting any attempt of manipulating votes:

- Voters digitally sign their votes which makes it impossible to change them.
- Voters enclose a proof of content knowledge (Schnorr signature) to prevent an attacker to create a new vote from the voter one.
- Digital signing takes place inside a trusted environment (digitally signed applet) on the voter’s machine.
- Voters are provided the possibility to review their votes before they cast and digitally sign them. Also, if a vote is made invalid by an attacker, the voting system will detect a failure and notify it to the voter.
- The voting platform checks that votes contain valid voting options before being accepted (ZKPs).
- The Mixing process is universal verifiable to detect any attempt by any of the Mix-net nodes to manipulate the votes.
- Voting receipts are provided to voters for verifying that their votes were not eliminated before reaching the e-vote counting phase.
- Ballot boxes are digitally signed and protected by immutable logs to detect any attempt to alter their contents

In addition, if return codes are used, the following features are added:

- Voters can check that their voting options are present in the vote recorded by the voting platform.
- Voters can check that no additional selections have been added to his/her cast vote.

Manual procedures are in place to prevent making unauthorised changes to p-votes.

<u>Elaboration of Requirement OS 8.10</u>
As part of the end-to-end verification features, the system allows voters to verify if their votes have been recorded as intended by means of using return codes.

During the election configuration process, unique vote cards are created for each voter. These vote cards will contain a unique Ballot Identifier (BID) and one Return Code for each possible option (party and/or candidate). One voter card only can be used by one voter to verify the proper treatment of one or more votes cast by the same voter. At the same time, one voter can use different voter cards for verifying different cast votes. However, the same voter card cannot be used by different voters to cast their votes to prevent coercion attacks (i.e., that a coercer gives the same vote card to different voters to verify their cast votes).

Once the vote is cast and the voter requests for verification, the verification process proceeds as follows:

- The voter sends as many encrypted proofs of selected voting options to the voting platform.
- The voting platform verifies that these proofs are correlated to the cast vote to ensure that the return codes will belong to the same stored options. This is done using ZKP protocols to prevent the disclosure of the voting options.
- Return Codes are calculated by an independent component of the voting platform (validation service) using the encrypted proofs.
- Return codes are sent to the voter so they can check that they are the same ones as on the vote card.

The return codes are related to the selected options (e.g., party, candidates) and the number of selected options (e.g., 1, 2). It is important to note that the code related to the number of selected options is generated by the voting platform without knowing how many options have been selected by the voter. The verification of the number of selected voting options prevents the inclusion by a rogue vote terminal of additional voting options without the voter's knowledge.

The return codes can be sent to the voter through the same communication channel used to cast the votes or an independent one. In the former case, only a voter selected subset of two digits of the return code will be returned to prevent any eavesdropper learning the complete code (the voter can use a different subset in further vote casting processes). In case an independent channel is used, the solution can be integrated with SMS or email gateways for sending the return codes using these channels.

The combination of the proposed protocol using return codes, voting receipts and universal verifiable mixing provides and end-to-end proof that the cast vote is received, recorded and counted as the voter intended.

The cryptographic details of the protocol used to generate the return codes can be found in the description of the protocol implemented in the solution (Appendix 2A Amendment 1).

<u>Elaboration of Requirement OS 8.12</u>
The decryption and counting of votes is implemented in an air-gapped environment: the Election Settlement Domain (ELSD). This environment is setup at the pre-election stage (configured with election configuration data, electoral roll, etc.), audited and sealed (physically and logically). After this processes the environment is stored in a secure place until the end of the election.

At the end of the election, the system can be started up and audited (to check any inconsistencies in the logical or physical seals). If everything is correct, the ballot boxes and any other election configuration updates are imported in the system using WORM media. The digital signature of the configuration files (if any) is verified before accepting them. The digital signature of the ballot boxes is also checked before imported. Any other data stored in the WORM is not processed nor executed by the system.

The decryption process consists on a Mixing cryptographic scheme. It cannot be initialized until the Electoral Board members introduce their shares to reconstruct the private key (or enable it in case an HSM approach is used). At the end of the process, the list of the clear-text votes is digitally signed and recorded in a new WORM media. The digitally signed list of voting receipt identifiers is also included. Counting can be implemented in the same air-gapped environment or a different one. The WORM media can be provided to auditors to implement parallel recounts.

<u>Elaboration of Requirement OS 8.13</u>
The annotation of the vote casting and the storage of the vote in the Electoral Roll are implemented as an atomic transaction by the system.

<u>Elaboration of Requirement OS 8.14</u>
In the case of an air-gapped system, the information must be digitally signed before being accepted (e.g., the Mixing Service requires that the Electoral Roll should be digitally signed by the board that administers the election).

In the case of online transfers/access to the information, at least the communication source should be authenticated (SSL). By default mutual authentication is required.

<u>Elaboration of Requirement OS 8.15</u>
The integrity of the data communicated in, to or from the Election System is maintained in the following ways:

- In air-gapped data transfers, the information is digitally signed before being exported to removable media. This is for example the case of the Election configuration information.
- In online transfers, the connection established is mutual authenticated using digital certificates (SSL). As an additional integrity level, the information can be also digitally signed before being transferred.

<u>Elaboration of Requirement OS 8.16</u>
The election configuration information must be digitally signed by an Election Administration board for their acceptance. Election system modules do not accept any information that has not been digitally signed by this Administration board. This board is formed the same way as the Electoral board (Multi-party computation secret sharing scheme) but its role is to validate any Election configuration change (i.e., custodies the integrity of the Election configuration).

Election related information that is processed (imported and exported) by the proposed solution is also protected by means of digital signature. This digital signature is done using the digital certificate of the source that exports this information. Depending on the information, we can distinguish the following sources and information samples (not only related to configuration information):

- Election services: digital certificates managed by the election applications (e.g., vote collector server, audit system, etc,)
    - Voting Receipts
    - E-vote Ballot Boxes
    - Immutable Logs
    - And any other election data exported by these services.
- Election officials: digital certificates assigned to individual election officials that are used to manage a specific election process
    - pVoting (OCR) results: (EML) signed by local election official in charge and scan operator
    - pVoting scanned ballots (TIF files): digitally signed by local election official in charge and scan operator.
- Election Board: Boards formed by members focused on ensuring the privacy and accuracy of election results (Electoral Board) or the integrity and authenticity of election information (Administration Board).
    - eVoting results: (EML) signed by minimum to Electoral Board members
    - Electoral roll: digitally signed by Administration Board.
- Auditors and certification authorities: External certification authorities, authentication authorities or PKI entities:
    - Application files: digital signature of files that should be executed by the application servers
    - CRL files: from any external CA
    - Authentication tokens or assertions: CAI

<u>Elaboration of Requirement OS 8.17</u>
Any modification of the system requires the validation (digital signature) of the Administration board. Any attempt to reset the system to an initial state will generate inconsistency checks and will cause the start-up to fail (e.g., the information of the immutable logs will not be coherent with the Election status).

<u>Elaboration of Requirement OS 8.18</u>
Votes are always encrypted on the voter terminal.

<u>Elaboration of Requirement OS 8.19</u>
The system provides means to allow voters to verify that they are connected to the official Election site, including:

- Verification of the SSL connection server: voters are able to check the authenticity of the voting front-end by verifying the digital certificate used by the server to establish the SSL connection (usually done by the web browser by default).
- Verification of the digital signature of the applet: voters can check the authenticity of the applet that executes the cryptographic protocol by checking its digital signature (usually done by the Java Virtual Machine by default)
- Verification of the digital signature of the voting receipt: in addition to the receipt identifier, the voter receives the value of the vote collector service digital signature of this receipt identifier. Voter can check this digital signature to verify if has been issued by the official election system
- Server return codes: The system incorporates a unique server return code that is provided to the voter with the voting card. If the voter is connected to the official election system, the proper code should be returned to the voter by the system after casting her vote.

In addition, the use of return codes allows voters to check if the voting options of the vote are aligned with the options of the return codes.

Elaboration of Requirement OS 9.1
System architecture has been designed in different layers that allow deploying of the system in high availability, scalable and heterogeneous environments:

- Voting front-ends:
  - Can be replicated and support automatic load balancing client sessions.
  - Implements load balancing connections against voting servers to automatically change the communication path in case one of the nodes is not responding.
- Voting servers:
  - Can be clustered to prevent the lost of service in case one node falls (active-active clustering)
  - Can receive load balanced connections from different voting front-ends.
  - Can receive connections from voting front-ends of different channels
  - Store the votes, logs and any critical information in a database cluster (e.g., Oracle RAC).

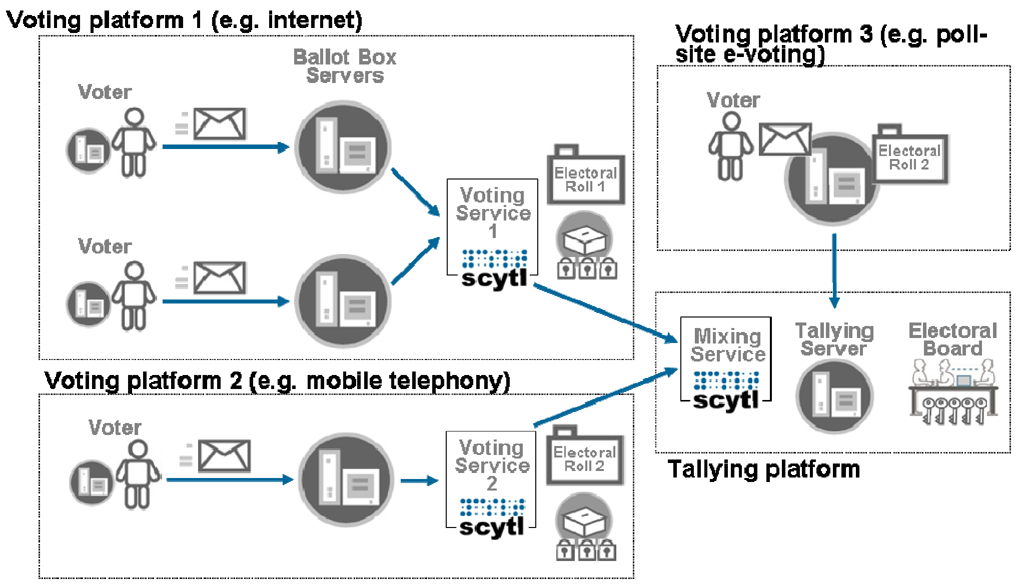


Figure – High availability, scalable and heterogeneous environments.

The system also provides deployment in redundant and fault tolerant hardware.

As part of the audit system, system checks of the e-voting services are implemented to check if all the nodes are working properly.

Elaboration of Requirement OS 9.2
The system does not implement large transactions that will prevent a quick restoration in case of a rollback if the system is accidentally restarted.

Elaboration of Requirement OS 9.4
As part of the audit system, the solution incorporates service checks that allow verifying the proper behaviour of the system components. These service checks are an end-to-end verification of all the components that the system requires to operate a service. For instance, the e-voting service check verifies that the front-ends, vote collector services and database service are properly working on a single check.

Elaboration of Requirement OS 9.5
The voter session is managed by a vote authentication token issued by tha authentication service component of the e-voting system. The authentication token contains information about the number of times the token can be used, in which contests can be used, expiration time and the time between casts. This information can be configured before the election to prevent DoS attacks.

Elaboration of Requirement OS 10.1
During the configuration of the election, the Election Administration board reviews and digitally signs the electoral roll and candidate list information. The digital signature on the electoral roll is verified by the voting platform before importing this information. Any tampering with the electoral roll would be detected before starting the voting or counting processes.

Elaboration of Requirement OS 11.5
In addition to perimitral measures such as firewalls and IDS, the system is designed in different layers that keep only the essential services exposed to public networks (e.g., voting front-ends). That allows the segmentation of the communication network between different layers on different physical sub networks. Connections between sub networks (and therefore components) can be restricted by means of firewalls.

System components can communicated between them (expect voting application) using IP address. Therefore, no DNS resolution is required,

In addition, the security architecture of the system includes provisions for not requiring the execution of the system components using privileged accounts. Furthermore, components are digitally signed to allow the verification of their integrity and authenticity before being executed.

To prevent the execution of malicious software on the server side, a three layer trusted chain of the server system is implemented. Please refer to OS 6.1 for the explanation of these three layers.

Elaboration of Requirement OS 13.2
The system components always check the integrity of the information managed by this service during the start-up process. Since this information is digitally signed (election configuration, immutable logs, etc.) these checks are completely reliable and accurate. In case a problem arises, the service notifies the operator and stops. The service is only available when it is restored to a stable state. These checks can be also performed by the election officers at any time during the election.

Problems with static data can be easily solved restoring this data (e.g., election configuration file). To solve problems with dynamic information (e.g., logs), tools are provided to restore the system to a stable state. Any action done to restore the system is logged.

The audit system receives all the logs from the system components and infrastructure. Monitors will be setup in the servers and audit system to detect and notify of any failure that could compromise the operation of any system. This includes:

- Monitor of system components: Server and infrastructure components (i.e., disks, network cards, routers, etc.) are monitored to detected any failure of the components and start a fast recovery procedure. Since all the critical components are redundant, this failure does not stops the service (it is still available through the redundant component). However, the failed component should be restored as soon as possible to prevent that a second failure could endanger the system operation.
- Monitor of the application: As part of the development, service checks are implemented in the applications. Service checks allow verification if the entire the stack of components required for the application is working properly. For instance, internal monitors set-up in the admin system to make an access to the web page used by voters and verify if it returns the expected results (e.g., that the credential provided is not valid) or an errors related to a component failure.

Elaboration of Requirement OS 13.3

The system generates enough information to allow reconstructing any voter transaction. This can be done without compromising the voter privacy, even if the information is provided to observers. Furthermore, all the information is digitally signed enclosing timestamps to allow verifying the integrity and authenticity of the information and therefore, implement meaningful audits.

In addition to the events registered from the operating system, database and infrastructure components (e.g., firewalls), the election application components (administration system, settlement, electoral roll, etc.) registers any action made by users related to election transactions. For instance, any change made by an election manager on a party candidate in the election administration system (during the election setup) is recorded. The information is recorded in a meaningful way, including at least the date, user, action, target of the action and the result (e.g., At DDMMYY HH:SS the election official X modified the candidate C name N to N1). The logged information allows retracing any actions made by users during a session. All transactions related to the access or modifications of securable objects are reported.

All this information is locally and remotely (audit system) stored in immutable logs.

All these practices facilitate the election observation.

Elaboration of Requirement OS 13.4

The extensive use of cryptography makes the audit system as part of the election system.
System is auditable at application, logical and technical level.

Elaboration of Requirement OS 13.5

The audit system of the election allows the recording in immutable logs of the information send from the different connected system sources. The received information is monitored and alarms send to responsible in case any suspicious behaviour or critical error happens. The audit system includes facilities for verifying the information and any cryptographic proof generated by the voting protocol.

Elaboration of Requirement OS 13.6

The audit system is based on standard cryptographic practices and is easy to understand (e.g., voting receipts). It also has provision that allows demonstrating the presence of any issue (e.g., voters can use the voting receipt to fill a claim in case their receipt identifier is not present in the verification list). It can also be integrated with standard log reporting mechanisms (syslog) for easy audit data gathering integration.

Every operating system, application, and device on the network generates log events to inform about current system status. The system collects, analyzes and correlates a copy of all these logs on the solution audit system to raise an alert in cause of attacks or errors.

The audit system has a set of agents (log analyzers) which are responsible for analyzing the different logs. Each agent is designed to work with a specific kind of logs: firewall, IDS, voting systems, etc.

Agents can be installed on the same server they are monitoring and/or installed in a centralized audit system in order to remove the load of analyzing logs from specific platform servers.

The audit system allows the Electoral Authorities to configure which incidents will raise an alert, allowing them to actively report and distinguish critical incidents over the regular noise on any system. Integration with SMTP, SMS, and SYSLOG allows the auditors to receive alerts through various channels, such as e-mail and handheld devices (e.g., cell phones and pagers). Every system agent allows configuring the events that raise alerts.
In addition, the audit system can integrate with current systems such as SIM/SEM (Security Incident Management/Security Events Management) products for centralized reporting and correlation of events.

Elaboration of Requirement OS 13.8

Logs, configuration information and election data have a standardized format (XML). The authenticity and integrity of these data can be verified by standard means (digital signature). These measures facilitate verifying if the election has been configured and implemented according the applicable legal provisions.

Elaboration of Requirement OS 13.9

Immutable logs and election data allow auditors to verify the integrity of the overall election. Additionally, the voting receipts allow voters to verify by themselves the correct treatment of their corresponding votes.
The audit system also includes all the verification processes implemented by the cryptographic protocol, allowing to check any election information generated during the complete voting and counting process: from the ZKPs of contents correctness issued by the voters when casting their votes, until the integrity proofs of the Mixing services and Electoral Board decryption proofs.

Elaboration of Requirement OS 13.10

Logs, configuration information and election data have a standardized format (XML). The authenticity and integrity of this data can be verified by standard means (digital signature). These measures facilitate to check if the election has been configured and implemented according to the applicable legal provisions. Additionally, the voting receipts allow voters to verify by themselves the correct treatment of their corresponding votes.

Elaboration of Requirement OS 13.11

The integrity and authenticity of the log registers and all the election data are cryptographically protected.

The techniques used for audit log data protection are:

- Log replication: all system and infrastructure component send a copy of their logs to the audit system. Therefore, in case the connection from the system or infrastructure component is broken, a local copy of the log is always available.
- Immutable logs: audit system protects the log information received using a cryptographic audit log. Therefore, any attempt to manipulate any single log entry is detected and the log information surrounding this entry isolated.
- Backups: Log information received by the audit system is periodically stored in backup media.
- Redundancy: The audit system implements two levels of redundancy:
    - Component level: can be setup in a redundant and fault tolerant server, with fault tolerant network paths that prevent a single point of failure
    - Service level: audit system can be configured in active-active fault tolerant mode. Therefore, several audit modules can be setup to receive log information from different systems at the same time. Load balancers can be setup in front of each audit server to redirect the connections from one server to the other in case of failure.

<u>Elaboration of Requirement OS 13.13</u>
In the worst case scenario, the integrity of the polling phase data can also be verified through the audit system. Since this system is independent from the rest. A failure of the system could not affect audit information and therefore, can be used to ensure the integrity of data.

The self tests and integrity checks performed prior to restart in the event of a crash/shutdown are:

- System status check
    - TPM checks to verify the integrity of the BIOS, boot loader and kernel.
    - Tripwire checks to verify the integrity of the system files
    - Internal operating system checks (e.g., file system checks, etc.)
- Database check
    - Verification of the data transaction status.
- Service check
    - Verification that the last two phase commit transaction finished properly (checking that was successfully committed by all the services) and rollback the transactions to a safe state in case some of the services did not finished it.
    - The service is checked end-to-end to verify that is properly working
- Data status check
    - Verifications that the last integrity proof of the stored information (e.g., votes of the ballot box) correspond to the last integrity proof of this information stored (immutable log).

Elaboration of Requirement OS 14.1
Supervision, auditability and certification are of utmost importance if the election administration and electronic voting solutions are to generate the highest levels of trust amongst all stakeholders. In this respect, the proposed solution provides the highest levels of trust currently available in the market thanks to the following measures:

- The source code can be audited by the customer and/or experts. After the audit, the solution components are digitally signed allowing the customer, auditors and observers to verify that the certified authentic solution components have been installed at the designated IT infrastructure and that they have not been modified.
- The digital signatures on the certified solution components proof that these components were audited and act accordingly to their specifications.
- The election configuration files are digitally signed by the Electoral Board (or an Administration Board on their behalf), allowing the customer, auditors and observers to also validate the integrity and authenticity of the election configuration.
- The digital envelopes inside the digital ballot box can be verified to ensure they were digitally signed by voters who are in the electoral roll. If a vote with an invalid digital certificate is detected, the system administrators will be alerted.
- Cryptographically chained logs are used to track the main actions carried out by system users. These logs cannot be tampered with without detection and therefore any wrong doing by any of the platform's privileged actors can be easily detected and proved.
- The review of components, binaries, logs, configuration files and other electoral data cannot affect in any way the integrity of the election and the privacy of the voters, as auditors do not have access to the required private keys.
- Voters can verify that their votes reached the electoral authorities, by means of voting receipt.

Auditors/examiners/observers do not need to have the keys used for signing the binaries and therefore cannot manipulate the solution without being detected. In the same way, the auditing tools can also be digitally signed and provided in a CDROM or similar media that ensures that nobody can modify such programs and that their integrity can be validated before being used. A bootable Linux live CD can be used to mitigate the risk of manipulation of underlying operating systems of the computers on which voting platform components are installed.

It is also possible to use tools to detect any modification at operating system level and other applications required by the solution. This way, periodic checks can be done to ensure authenticity and integrity.

In addition, auditors can use the "TPM – Tripwire – application digital signature" system integrity chain to verify the integrity of the system that is executing the audit code.

Elaboration of Requirement OS 14.4
All system components of the voting system are certifiable. The scope of the certification process may vary depending on specific requirements per election concerned. The minimum perimeter is the following (not exhaustive):

- Compliance with legislation and requirements

- Integrity of election results and accuracy
- Secrecy of the vote
- Integrity of the vote
- Confidentiality of vote
- 1 voter = 1 vote
- Audit ability of the solution
- End-to-end application level security, from the voter to the Electoral Board
- The source code of the solution has no malicious code
- The solution meets the requirements in terms of usability and accessibility
- The cryptographic applications have been implemented according to recognized standards
- The system implements strong security measures based on cryptography
- Data integrity has been protected
- The integrity of system has been properly protected
- The confidentiality and integrity of communication channels have been properly protected

ErgoGroup/Scytl will provide the source code of the solution to the Ministry for audit and certification purposes. The main steps of the certification process, as done also in other countries, can be as follows:

1. ErgoGroup/Scytl deliver the source code plus required documentation to the customer: implementation guides, source code documents, diagrams, compilation and build guidelines, etc. The solution uses open source third party components for which the source code is also available for audit. If a Linux operating system is used, the source code for that is available as well. The source code can be provided in a virtual machine where the compilation and build environment has been set up. The build environment and other tools used can also be open source allowing maximum audit possibilities. A digital signature (or fingerprint) of the source code and other components is provided.

2. The customer and/or a group of experts selected by the customer validate the digital signature and review and audit the source code. ErgoGroup/Scytl can provide support to this group, for example by introducing them on the system and the documentation provided. The scope of the audit and the responsibilities of the parties involved are defined upfront in a statement of work.

3. Any discovery done by the customer and/or the group of experts is notified to ErgoGroup/Scytl, who will agree with the customer about how and when to fix any issues that were found. Usually these are fixed immediately after being detected. The group of experts can then receive a new version of the source code and verify that the flaw has been solved.

4. The customer and/or the group of experts digitally sign (or create a fingerprint) the audited and certified source code and other relevant components. They set up the compilation and build environment or use the one provided by ErgoGroup/Scytl. Using a trusted build they create the solution binaries, which are also

digitally signed. Independent auditors can supervise and certify the compilation and build process to assure that the digitally signed binaries are created from the digitally signed source code.

5. The digitally signed binaries are installed at the designated IT infrastructure. Independent auditors can supervise and certify this process.

6. At all times, the customer, election observers and auditors can validate the digital signatures of the solution binaries against the binaries installed at the designated IT infrastructure, ensuring that these binaries are those audited by the Ministry and/or the group of experts.

7. Auditors/examiners/observers do not need to have the keys used for signing the binaries. Auditing tools can also be digitally signed and provided in a CDROM or similar media that ensures that nobody can modify them and that their integrity can be validated before being used. A bootable standard Linux live CD can be used to mitigate the risk of manipulation of underlying operating systems, drivers, etc.

Please note that the Election configuration is also certified and digitally signed by the Electoral Board (or an administration board on their behalf) allowing to also validate the integrity and authenticity of the election.

The Voting Client Java applet is digitally signed after being approved by the customer and/or the group of experts. When downloaded (in a transparent way) by the voter to his/her computer, the digital signature is checked and a popup window appears in front of the voter asking for confirmation whether to accept or not the downloaded applet. This includes several data about the applet, such as the application's name, the provider, the certification authority that digitally signs it and the integrity status. Therefore, any part, including the customer and the citizens, can easily and in a transparent way verify the version and integrity of the Voting Client downloaded from the server. To be sure that the voter downloads the correct Voting Client, the connection to the server will use SSL, which also ensures the authenticity of the server.

Since part of the audit process is compliance with legislation and requirements, once the audited solution is certified means that the system components act accordingly to their specifications. The digital signatures on the certified solution provide proof for this.

In addition, cryptographically chained logs are used to track the main actions carried out. These logs cannot be tampered with without detection and therefore any wrong doing by any of the platform's privileged actors can be easily detected and proved. These logs allow the Ministry and/or auditors and observers to validate that the system is functioning correctly.

User friendly monitoring tools are provided that, together with an adequate monitoring policy, allow the customer, auditors and election observers to very transparently observe and monitor the status of the elections and system components in real time. With video cameras it is furthermore possible to observe used IT architecture. Observation possibilities could be setup in one or more election observation rooms or it could be provided to the public, for example by using the Internet.

6. Elaboration of External Interface Requirements

<u>Elaboration of Requirement EIS 4</u>
The system will authenticate users in the CAI (DIFI or Altinn) using SAML 2.0 over SOAP and HTTP. The protocol for authentication is as follows:

- User accesses the Election System application.
- If the user has no valid session, she will be transferred to the CAI using a HTTP redirect.
- The CAI displays a logon form and authenticates the user.
- The CAI redirects the user back to the Election System, providing a SAML artifact identifying the user.
- The Election System requests a SAML verification of this identity in the CAI using SOAP.
- The Election System performs queries in the CAI to retrieve social security number (using SOAP/SAML), and other information that may be relevant.

<u>Elaboration of Requirement EIS 5</u>
The election system can, as an option, interface with Altinn for the submission and processing of list proposals. In this configuration Altinn becomes the primary user interface. Altinn interacts with the Election system through web services exposed to the Altinn Service Engine.

The Altinn user interface provides the following functionality for parties and officials:

- Select relevant election or referendum.
- Select party from the Central Coordinating Register for Legal Entities – "Enhetsregisteret".
- Request for approval to create a new party, with upload of scanned signature file.
- Enter or upload party candidate information for approved parties, with the functionality to add, change, or remove candidates.
- Notification to approvers and manage changes/approvals for these.
- Feedback to parties and candidates regarding approvals and changes.

The Election system provides the following web service functionality for Altinn:

- Query election or referendum details.
- Submit scanned lists for ICR processing.
- Check of duplicate entries among other party lists.
- Verification of candidates in Electoral Roll (if not provided by Altinn).
- Submit approved lists and queries to retrieve approved lists.

<u>Elaboration of Requirement EIS6</u>
The Election system provides functionality to import structured information on count results provided to municipalities by other vendor software. An upload screen, available to authorized users in the municipalities, provides functionality to enter election information and import a file with counts. Standard data format is EML V4.0 Count (510). The file can be imported multiple times during the allowable timeframe for the election.

Elaboration of Requirement EIH1
The Election System provides an application for electronic counting of p-votes. This application uses OCR software from ReadSoft, which supports ISIS compliant scanners. The list of certified scanners includes models from Canon, HP, Fujitsu, Kodak and others.

Elaboration of Requirement EIH2
The reporting module generates polling cards, voting cards, etc. based on standard or custom templates. The result can be exported as PDF. Polling stations can print polling or voting cards locally with installed PDF reader software. It is also possible to export files for transfer to an authorized print service operator.

7. Elaboration of Documentation Requirements

Elaboration of Requirement D 2
The following documentation is povided as part of the Technical system documentation:

Guideance documentation:

1. **A system development guideline document**. This document describes the tools used for source control, building of code, programing guidelines and naming conventions.
2. **A configuration control guideline document**. This document describes the main principles in configration control. This applies to source-code, scripts, configuration files and so on. The documnt also describes the main prinsiples for deployment and staging the solution through the different test envirmnents.
3. **A quality control guideline document**. This documnts describes the qality control effort and the procedures that should be followed by the designer, developers and testers.
4. **A security policy document**. This document describes the security policy that the solution is build upon. The security policy also includes guidelines for the developers.

Design documentation:

5. **An overall system design document**. This documnts describes the overall system architecture and the different sub-systems and their main modules. The document also describes the main dataflow on all interfaces both internally and externally. The document consist of two main sections:
    - Product Specification Document
        - Success criteria
        - Functional requirements
        - Non functional requirements
        - Initial project risks
        - Functional description
    - Solution Design
        - Key concepts
        - Architecture
        - Components
6. **Module design documentation**. The documentation is developed as part of the development process of each sub-system or module. It is ”as buildt” documentation. It contains:
    - Responsibilities
    - Security requirements
    - Implementation details
    - GUI Design

- Inputs
- Process description
- Outputs

7. **Detailed security description document**. This document describes all mechanisms, protocols, keys and so on that are implemented in the system to meet the security objectives.

Vulnerability Assessment Documentation:

8. **A risk analysis document**. This document provides a risk analysis and with reference to the detailed security document, describing the mechanisms that are implemented to meet the security objectives set forth in the system.

Test documentation

9. **Test reports from the contractors system test**
10. **Test guidelines and tools for regression test.** This document is part of the lifecycle documentation and helps the application management process to do effective regression test after changes have been applied.

Elaboration of Requirement D 3
The following documentation is provides for the installation and operations:

1. **An installation preparation document**. This document describes all the condition that must be fulfilled prior to installation of the system. This includes requirements for hardware, system software, and 3'rd party SW. The document also gives a detailed description of the configuration needed on each system component. The document also gives guidelines at the requirement level for setting up proxy-servers, load-balancing and so on.
2. **A configuration and deployment guide**. This document describes how official releases of the system may be extracted from the configuration control system and how each module is to be deployed. It also describes how previous version may be deployed in the case of roll back.
3. **The installation and verification guide**. This document describes the overall installation procedures-However, as part of our quality control procedures each official release is followed by a step-by-step description ion how that specific release should be installed and verified. Obviously each of our contracted deliveries (refer appendix 4 for each delivery) will have the detailed installation description.
4. **The operational guide.** This document describes the application operational procedures. It includes check lists for the day to day operations, description of scheduled or manual procedures, common error situations and how they should be handled / investigated. It also provides performance and scaling guidelines.

Elaboration of Requirement D 4
The following user documentation is provided:

1. **A functional overview document.** It describes the system and the different modules and sub-systems from a functional point of view.
2. **A system administrator guide.** This document describes the different functions from a system administrator perspective. It also gives guidelines for the key-management functions provided in the system.
3. **An election configuration guide.** This document describes the overall data model of the system. It gives a detailed description on how to configure contest, elections and election events. It also describes the templates supporting the current Norwegian elections and how they can be parameterized.

4. **A reporting guide.** This document describes the general concept of the report generator and the predefined reports developed that support the official reports that are to be produced.
5. **An election administration guide.** This document describes the functions that are relevant for the electoral board.

Online context sensitive help
Help screens for all applications screens will be provided in the user's selected language. The help texts will be stored as Java Resource Bundles and loaded with the application to support Internationalization (i18n).

The level of context sensitive help will be designed as needed: a general help screen will be provided for each screen with a general description of functionality and field-level help. Particularly difficult sections will have several tool-tip buttons placed logically next to the input fields.