

# Requirements Engineering for E-Voting Systems

Kevin Daimi, Katherine Snyder, and Robert James  
 Department of Mathematics and Computer Science  
 University of Detroit Mercy,  
 Detroit, Michigan 48219, USA  
 {daimikj, snyderke, jamesr}@udmercy.edu

## ABSTRACT

Manual voting systems have been deployed for many years with enormous success. If those systems were to be replaced with Electronic Voting Systems, we have to be absolutely sure that they will perform at least as efficient as the traditional voting systems. Failures or flaws in Online Voting Systems will jeopardize Democracy in the country implementing them. The main focus of requirements engineering is on defining and describing what a software system should do to satisfy the informal requirements provided by a statement of need. In this paper, we will define and describe what the Online Voting System should do to ensure a robust, accurate, secure and quality-based design and implementation.

## Keywords

*Online Voting Systems, Requirements Engineering, Functional Requirements, Security Requirements, Nonfunctional Requirements, Use cases.*

## I. INTRODUCTION

There has been a great debate on the advantages and problems of various electronic voting schemes. Questions like “How the internet will alter U.S. democratic institutions,” “How the Americans will get information about the elections,” and “How Americans would vote in general elections,” have encapsulated the attention of many Americans. The prospect of being able to vote “in your pajamas,” as it is being described, captured the imagination of political leaders, technology innovators, and voters around the world [11].

The aim of electronic voting schemes is to provide a set of protocols that allow voters to cast ballots while a

group of authorities collect votes and output the final tally [1]. Problems with voting machines extend from the quality of the locks, to the need for a printed audit trail, to the hacking of the communication links [16]. Although voting makes many people to believe that voting is the perfect application for technology, but in reality applying it is hard. For a voting system to be ideal, four attributes must be satisfied: anonymity, scalability, speed, and accuracy [19]. These attributes will be covered by both the functional and non-functional requirements.

Requirements are defined during the early stages of system development as a specification of what should be implemented. They describe how the system should behave or a system attribute [21]. In other words, they represent what the system should do from the stakeholders’ point of view, and they should meet their needs.

There are a number of ways to define requirements engineering. Requirements engineering is the first major activity following the completion of a statement of need. It is defined in terms of its major activities: understanding problems, solution determination, and specification of a solution that is testable, understandable, maintainable, and that satisfies project quality rules [13].

A number of researchers relate requirement engineering to some goals. Requirements Engineering (RE) is concerned with the identification of goals to be achieved by the envisioned system, the refinement of such goals and operationalization into specifications and constraints, and the assignment of responsibilities for the resulting requirements to agents such as humans, devices, and software [8]. According to Nuseibeh and Easterbrook [12], Requirements Engineering is the branch of software engineering concerned with the real-

world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families. Goal-oriented Requirements Engineering is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analyzing, negotiating, documentation, and modifying requirements [9].

One of the main objectives of Requirements Engineering (RE) is to improve systems modeling and analysis capabilities so that organizations can better understand critical system aspects before they actually build the system [17]. The functional requirements along with quality attributes and other nonfunctional requirements will constitute the Software Requirements Specification [22]. Functional requirements are the capabilities of the system and domain specific. However, nonfunctional requirements are constraints on the functional requirements or quality requirements.

There are a number of techniques to modeling, representing, and checking requirements. Some of these approaches are; Case-Driven, Viewpoint-Based, Behavioral Pattern Analysis (BPA), Software Architecture Orientation, and Formal Methods approaches [14], [10], [4], [6], and [20].

In this paper, both functional and nonfunctional requirements for Online Voting Systems are presented. They will describe how an online voting system ought to behave. For a system that can have a great impact on democracy and the way Americans will vote, engineering the requirements is crucial as no one will trust a system that is constructed based on wrong or imprecise requirements. As the design and implementation of Online Voting Systems has requirements engineering as its foundation, we need requirements that have zero tolerance with respect to deviating from actual need. This paper also emphasizes the need for voting system security requirements. Example of use cases will be provided.

## 2. USER GROUPS

The key to successfully using the online voting system is the ability to use the system and access the information available to help. The help facility should be fully functional and able to instruct users through every step while allowing others more versatility in using the web environment. This is achieved by skipping all help functions and proceeding directly to the voting process. Accordingly, users are divided into the following six groups:

1. *Knowledgeable Group*: We believe the more educated the person is, the less likely the help function will be needed and the probability of successfully completing the voting is high.
2. *Frequent Group*: These are users that surf the web frequently for various purposes. In general they perform routine tasks. Most of them have memorized the steps needed to get to the site they need. However, it does not necessarily mean they can use the online voting system without any problem.
3. *Inexperienced Group*: This group of users includes those who use the web very rarely or not at all. They will, most likely, need more assistance and, therefore, need more time in carrying out the voting process. This group of users will have a high number of elderly.
4. *Government Group*: This group will be mainly using the administration functions needed for counting and maintaining the voting data. The group will also be involved with setting up and completing the ballots for regular users.
5. *Technical Group*: This group will be in charge of troubleshooting and maintaining the software, hardware and the network. They will not have access to actual voting data.
6. *Computer and Network Security Group*: As security is essential for such a system, this group will ensure that security is met at the software, hardware, network and physical levels.

## 3. PROBLEM-SOLUTION CHARACTERISTICS

There are a number of problems that the online voting solution should address. Among these are the following:

- Voter secrecy: No one should know what the voter voted
- Voter authentication: Voters should be who they claim they are
- Verifiability of votes: Internal tracking of votes, to assure all ballots are registered to a voter.
- Accuracy of voter turnout. Each voter is tracked to completion, so voter data is available at any time.
- Safe transfer of votes from user's computer to the server
- Safety of caste votes: Proper security process and user registration can guarantee ballot assurance.

- Uniqueness of casting – A person can cast only one vote
- Permitting the voter to vote for as many candidates for an office as the voter is lawfully entitled to vote for without exceeding the limit
- Empty ballot box at the start of voting
- Voter should be able to verify the vote before it is cast
- Provision for editing the vote any number of times
- User manuals should be provided for voters several days before election
- Trial version should be released several days before the election
- All server operations, whether operating system function, software functionality or OSI (Open System Interconnection) model functionality, must be protected

The above mentioned problems will give rise to the question of economical benefits of the online voting system (solution). Once the product is released, it should have the following benefits:

- (1) If the online voting system is successful, people need not go to the polling booths to cast their votes. They can vote from their home and hence a lot of time will be saved
- (2) The existing paper ballot system will be discarded and hence a lot of materials can be saved
- (3) Counting the ballots will be executed more accurately, quickly, and consistently
- (4) As the existing paper ballot system will be discarded, many resources deployed by the Government will be freed for other purposes
- (5) Reports can quickly be generated and hence a lot of manual labor will be saved

The output of a voting system is characterized as good if it is capable of verifying the votes, providing accuracy of the voter turnout to the number of people voted, avoiding coercion, and counting all votes.

#### 4. FUNCTIONAL REQUIREMENTS

Enhancement to the online voting system will primarily provide a more precise vote management tool that will establish accountability and improve data accuracy, and thus allowing voters to feel a greater level of confidence in the reported data. The majority of the precinct managers, who will benefit from these enhancements, currently use their professional judgment and expertise to anticipate the voter's needs when making decisions.

They also rely on outside vendor data and poorly captured metrics from the current state of traditional voting system.

Appropriate behavior constitutes the functionality of a system and there is often a tight correspondence between particular requirements and particular functions of the solution system [2]. The following represents a partial list of functional requirements for the Online Voting System:

- The system must provide voters with accurate data
- Metric reports of current/live votes must be provided
- The system should make use of tools available for users on the internet
- It must adhere to government requirements
- Ease of GUI use that can be accessed via web browser must be established
- The system must follow technical development standards supported on known operating systems such as Windows, Linux, and UNIX, in addition to future operating systems versions
- The system must grant technician/customer general communications and training documents
- The system must supply a prototype or process to approve site customization
- Backup data restore capabilities should be granted
- The system must send a notification to administrator if an onsite workstation is classified as inoperative or unusable
- The system should send a notification to administrator of updates from verification popup windows
- The system must supply standard reports for decision making
- Audit trails of who made changes to the database must be maintained
- The system should allow voting administrators to make updates to the voter information database
- The system must verify on a daily basis responsible users ID and location
- The system must provide standard error checking
- The system must provide data integrity checks to ensure data remains consistent and updated

#### 5. MAJOR CONSTRAINTS

When dealing with requirements engineering for any systems, there are some constraints that must be

considered. The major constraints for the Online Voting System are:

1. Voting is carried out from many consoles on the internet.
2. All voting is done in one day.
3. Many interfaces exist including Windows Explorer, Netscape, and Mozilla browsers.
4. The operating system in use are, but not limited to, Windows, Linux, and UNIX.
5. Many different levels of expertise in the system use will be prevalent.
6. Each state can administer the system differently depending on state laws.
7. Each state can have unique election and proposals, needing many different administrative interfaces.

## 6. NONFUNCTIONAL REQUIREMENTS

Nonfunctional requirements are requirements that are not specifically concerned with the functionality of a system. They normally place restrictions on the product being developed and the development process [7]. Nonfunctional requirements may be regarded as parameters of functionality in that they determine how quickly, how accurately, how reliably, how securely, etc., functions must operate [2]. Some of the Online Voting System's nonfunctional requirements are as follows:

- Response and net processing time must be acceptable by user and by application
- Defects in the local voting database file must be less than **0.0001%**
- Defects contained in the collection server must be less than **0.0001%**
- Defects in the database must be less than **0.0001%**
- Number of collection failures per voting process must be at **six** sigma, or better. Taking the entire state of Michigan, with a population of about 9.9M citizens (if all voted), as an Example, this works out to be **33** errors
- **When checking the database for errors, a 100% scan of the data is required, rather than selecting a sample set.**
- The system must be working at 100% peak efficiency during the voting process.
- Transfer of existing and future data to an Voting Management Data Center must be granted
- The system should be allow adding more voters to allow a greater connectivity rate

- A process must be devised to support normal precinct business hours
- The system should support response time for **addressing severe issues in less than 5 minutes**, due to the shortness of the voting timeframe.
- The system should provide documentation to inform users of system functionality and any change to the system
- The system should provide friendly graphical Interface to ensure ease of use when end users utilize system functionality

## 7. SECURITY REQUIREMENTS

Electronic voting systems represent a great security challenge. Any successful attack would be highly visible, and thus, motivating much of the related hacking activity to date [18]. Traditionally, security is incorporated in a software system after all the functional requirements have been addressed. Due to its criticality, security should be integrated in the software life cycle [5]. Voting software security can be achieved if security is merged into voting software functional requirements during the early stages of software requirements engineering [3].

Although, security requirements are nonfunctional requirements, we deliberately avoided including them within the nonfunctional requirements due to the crucial role they play in the success of the any online voting system. Below is a partial list of the Online Voting System security requirements.

- The voting system should include controls to prevent deliberate or accidental attempts to replace code such as unbounded arrays and strings
- The system should have zero-tolerant with regard to compromising
- Election process should not be subject to any manipulation including even a single vote manipulation
- The system should provide accurate time and date settings
- The system should not allow improper actions by voters and election officials
- The system should not allow Local Election Officials (LEOs) to download votes to infer how voters in their precinct have voted
- The system should provide means for protecting and securing recounts of ballots cast in elections

- The system should not allow voter submissions to be observed or recorded in any way that is traceable to the individual voter
- The system should ensure that election results would be verifiable to independent observers. This implies that published election results correspond to the ballots cast by legitimate voters
- The system should not allow tampering with audit logs

10. The voter is allowed to edit his/her vote any number of times.
11. If she/he is satisfied with the final vote screen, he/she casts the vote.
12. If the vote reaches the server, a message is displayed to the voter that his vote has reached the ballot.
13. The voter logs out.

**Exceptions:**

1. The voter may enter the wrong details.
2. The voter might try to select options more than the allowable ones.
3. The voter's connection with the server may terminate before the vote reaches the server.
4. The voter's connection with the server may terminate in the course of the session.
5. After the vote is cast, the voter may try to navigate back to cast another vote.

**Event:**

If the voter is not identified in three attempts, the process stops and the voter needs to contact the election conducting authority to restart the process.

**Frequency:**

Used as many number of times as there are voters.

**Secondary Actors:** Election conducting staff who are contacted by voters in case of difficulties.

## 8. DEVELOPING USE-CASES

A use-case tells a stylized story about how an end-user interacts with the system under a specific set of circumstances. The story may be narrative text, an outline of tasks or interactions, a template-based description, or a diagrammatic representation [15]. Regardless of its form, a use-case depicts the system from the end-user's point of view. Examples of use-cases for the Online Voting System are given below.

### *USE CASE 1: Voting*

**Actor:** Any person that is allowed to vote

**Goal:** To cast their votes in a safe and secure manner.

**Preconditions:** The process is password protected. The voter must know her/his PIN, without which they cannot vote.

**Scenario:**

1. The voter enters the website address in his browser.
2. The voter selects the state to which he/she belongs.
3. The user is allowed to have a look at the tutorial section which is optional.
4. The voter enters the Name, SSN, State ID, Date of Birth, and Gender.
5. If the input of the voter matches the records, he/she is allowed to login.
6. The voter is allowed to choose one of two options: Party Selection or Individual Selection.
7. The voter casts her/his vote to their favorite choice under a selection.
8. The voter navigates to all the pages and votes to his/her choice under each category.
9. The voter checks the final screen of the vote.

### *USE CASE 2: Configuration*

**Actor:** Configurator (usually an authorized person of the election commission).

**Goal:** To configure the voting system by entering the offices for which voting is to be done and configuring the candidates for the offices.

**Preconditions:** There are no preconditions while installing.

**Scenario:**

1. The actor clicks the button "Configure."
2. The actor clicks either "Single Configuration" or "Multiple Configuration" button based on whether the election is held for a single province or a multiple province.
3. If the "Multiple Configuration" button is pressed, the actor is prompted to enter the common

offices and the offices that are specific to that province.

4. The actor enters the criteria based on which provinces are distinguished.
5. The actor is allowed to add a new office or edit an existing office by pressing “Add New Office” or “Edit Existing” button respectively.
6. The name of the office and the number of candidates for that office are entered.
7. The actor clicks the next button which allows him/her to enter the name of the candidates and the party to which they belong.

**Exceptions:** There are no exceptions.

**Frequency:** Usually once.

**Secondary actors:** Software staff.

## 9. CONCLUSIONS

Voting might look like a suitable or perfect choice for computer applications, but in reality implementing it is harder than it first appears. Many comments have been made by computer professionals and voting officials on electronic voting systems advantages and disadvantages with a special emphasis on their security.

We have attempted to tackle the problem of representing stakeholders’ needs for an online voting system. Essential functional requirements, that lay the basis for the system design phase, have been stated. These were supported by nonfunctional requirements including security requirements. The Requirements Engineering for E-Voting should be flaw-free. It is our belief that flaws in an online voting system will result in failure of the voting system which will jeopardize democracy and disappoint voters.

## 6. References

- [1] O. Baudron, P. Fouque, D. Pointcheval, J. Stern, and G. Poupard, “Practical Multi-Candidate Election System,” In Proc. The Twentieth Annual ACM Symposium on Principles of Distributed Computing, 2001, Rhode Island, USA, pp. 274 - 283.
- [2] I. Bray, An Introduction to Requirement Engineering. Harlow Essex: Addison Wesley, 2002.
- [3] K. Daimi, and C. Wilson, “Electronic Voting System Security requirements Engineering,” in Proc. The International Conference on Software Engineering Research and Practice 2005, Las Vegas, USA, pp. 230-235.
- [4] A. El-Ansary, “Behavioral Pattern Analysis: Towards a New Representation of Systems Requirements Based on Actions and Events,” In Proc: SAC, 2002, pp.984-991.
- [5] D. P. Gilliam, T. L. Wolfe, J. S. Sherif, and M. Bishop, “Software Security Checklist for the Software Life Cycle,” in Proc. WETICE’03, 2003, pp. 243-248.
- [6] J. H. Hausmann, R. Heckel, and G. Taentzer, “Detection of Conflicting Functional Requirements in a Use Case-Driven Approach,” In Proc: ICSE’02, 2002, pp. 105-115.
- [7] G. Kotonya, and I. Sommerville, Requirements Engineering – Processes and Techniques. Chichester, West Sussex, England: Wiley, 2003.
- [8] A. van Lamsweerde, “Requirements Engineering in the Year 00: A Research Perspective,” In Proc. ICSE’2000: 22<sup>nd</sup> International Conference on Software Engineering, 2000, pp. 5-19.
- [9] A. van Lamsweerde, “Goal-Oriented Requirements Engineering: A Guided Tour,” In Proc. The 5<sup>th</sup> IEEE International Symposium on Requirements Engineering, 2001, 249-263.
- [10] H. Mei, “A Complementary Approach to Requirements Engineering – Software Architecture Orientation,” Software Engineering Notes, Vol. 25, No. 2, pp. 40-47, March 2000.
- [11] J. Mohen, and J. Glidden, “The Case for Internet Voting,” CACM, Vol. 44, No. 1, pp. 72-85, Jan. 2001.
- [12] B. Nuseibeh, and S. Easterbrook, “Requirements Engineering: A Roadmap,” In Proc. The International Conference on Future of Software Engineering, 2000, pp. 35-46.
- [13] J. Peters, and W. Pedrycz, Software Engineering – An Engineering Approach. New York, NY: Wiley, 2000.
- [14] J. van der Poll, and P. Kotzé, “Combining UCMs and Formal Methods for Representing and Checking the Validity of Scenarios as User Requirements,” In Proc: SAICSIT, 2003, pp. 59-68.
- [15] R. S. Pressman, Software Engineering: A Practitioner’s Approach. New York NY: Addison Wesley, 2005.
- [16] J. Raksin, “The GIGO Principle and Voting Machine,” ACM QUEUE, Vol. 2, No. 2, pp. 10-11, April, 2004.
- [17] W. Robinson , S. Pawlowski, and V. Volkov, “Requirements Interaction Management,” ACM Computing Surveys, Vol. 35, No. 2, pp. 132-190, June, 2003.
- [18] A. D. Rubin, “Security Considerations for Remote Electronic Voting,” CACM, vol. 45, pp. 39-44, Dec. 2002.

- [19] B. Schneier, "Voting Security and Technology," IEEE Security & Privacy, Vol. 2, No. 1, pp 10-10, Jan. 2004.
- [20] A. Silva, "Requirements, Domain, and Specification: A Viewpoint-Based Approach to Requirements Engineering," In Proc. ICSE'02, 2002, pp. 94-104.
- [21] I. Sommerville, and P. Sawyer, Requirements Engineering – A Good Practice Guide. Chichester, West Essex, UK: Wiley, 1997.
- [22] K. E. Weigers, "Karl Wieggers Describes 10 Requirements Traps to Avoid," Software Testing and Quality engineering, Vol. 2, No. 1, pp. 34-40, Jan./Feb., 2000.