Available online at www.sciencedirect.com
ScienceDirect

Computer Standards & Interfaces 29 (2007) 376–386

COMPUTER STANDARDS & INTERFACES

www.elsevier.com/locate/csi

Electronic voting systems: Requirements, design, and implementation

Ghassan Z. Qadah [*], Rani Taha

Computer Engineering Department, American University of Sharjah, Sharjah, United Arab Emirates

Abstract

The term “Electronic Voting (e-voting)” refers to the use of computers or computerized equipment to cast votes in an election. e-voting aims at increasing participation, lowering the costs of running elections and improving the accuracy of the results. This paper details the requirements, design and implementation of a special type of electronic voting systems, the remote on-line voting system, suitable for university setting where students can cast their votes anytime, anywhere and using fixed and mobile electronic devices including personal computers, personal digital assistants and smart and regular phones. To avoid web content replication for each of the connecting devices, the implemented system separates the data content from its presentation form. The separation is achieved by using modern technologies such as the extensible markup language to represent the web data content and the extensible style language transformation style sheets to customize the presentation of such content on different connecting devices, thus, achieving true “author once, publish to any device” design and implementation.

Keywords: Database systems; Electronic voting; Mobile devices; Web-based application; Wireless network interfacing; **XML** data representation

1. Introduction

Voting and elections are essential ingredients of modern communities. Unlike any other transactional event, the result of elections can have many positive and/or negative effects on these communities and their wellbeing. For many years, elections, in general, have suffered from declining participation rates due to the inconvenience of manual voting. Manual voting has several other drawbacks among which are inaccuracy in ballot counting and the delayed announcement of election results [1]. To overcome these drawbacks, the Electronic Voting (e-voting) technique, the use of computers or computerized equipment to cast votes in elections, has been proposed. e-voting automates and simplifies the election process, speeds it up, increases participation rates, reduces counting mistakes and minimizes the time it takes to announce results.

This paper details the requirements, design and implementation of a generic e-voting system capable of supporting a number of election campaigns simultaneously and permits voters to cast their votes in these elections at anytime (anytime voting), from any-location (anywhere voting) and using a variety of electronic devices (any-device voting) including Web browsers, **WAP**-enabled [9,10] devices (Personal Digital Assistants (**PDA**s), Smart Phones) and **SMS**-based [11,12] mobile phones. To avoid web content replication for each of the connecting devices, the implemented system separates the data content from its presentation form. To achieve such a separation, the Extensible Markup Language (**XML**) [13] is used to represent the web data content, whereas, the Extensible Style Language Transformation (**XSLT**) [14] style sheets are used to customize the presentation of such a content on different connecting devices, thus, achieving true “author once, publish to any device” design and implementation. To validate this concept, this paper presents the implementation of a prototype for the e-voting system in support of student elections within a university setting.

The rest of this paper is organized as follows. Section 2 presents some background material on e-voting systems. The relevance of the research reported in this paper to the general area of computer interfaces and standards is also presented in Section 2. Section 3 details a set of high-level requirements that an e-voting system must satisfy. The technologies and standards needed to support the development and implementation of e-voting systems are presented in Section 4. Section 5 outlines the implementation of a prototype for an e-voting

[*] Corresponding author.
E-mail address: gqadah@aus.edu (G.Z. Qadah).

0920-5489/$ - see front matter © 2006 Published by Elsevier B.V.
doi:10.1016/j.csi.2006.06.001

system using the requirements and technologies outlined in Sections 3 and 4, respectively. Finally, Section 6 presents some concluding remarks.

2. Background

2.1. Related work

Elections, in general, can be divided into two main types, namely, political (e.g. legislative and municipal types of elections) and non-political (e.g. election of student governing body within a university, election of trade associations). Both of these election types have different security requirements. While high security is needed by political elections, a substantially less amount of security is needed by the non-political ones. Different electronic systems have been proposed in support of elections and voting, namely:

1. **Computer Counting**: this type of systems requires the voters to mark their votes on a paper with a pencil or marker or remove divots from a perforated card [1]. The ballot cards are then scanned and tallied at a central computer site or at each polling station.
2. **Direct-recoding electronic voting machine (DRE)**: this system requires voters to use a keyboard or touch screen to mark their votes on a computer terminal, directly connected to a stand-alone, polling-station-located computer [2]. The votes are immediately added to a running tally stored in the computer's storage system. The final **DRE** tally is then moved to a central location where it is added to the tallies obtained from other **DRE** machines. A comparison between **DRE** and paper-based voting systems is presented in Ref. [3].
3. **On-line Voting**: this type of systems involves the use of a computer and the Internet or a private network in support of the voting process. On-line voting can be conducted in a variety of ways, namely [1]:

- "**Poll site" e-voting system**: this type of voting systems requires voters to go to staffed polling sites and use computers to cast their votes. A network (Internet or private) is used to transfer ballots from each polling place to a centralized site, where votes are tallied and election results are published.
- **"Kiosk" e-voting system**: this type of systems allows voters to vote from computers/ATM-like machines situated within kiosks. The kiosks are setup by the voting authority in suitable locations such as post offices or shopping malls and connected to a central location via the Internet (or a private network). A vote casted at the kiosk will immediately be forwarded across the network to the centralized tallying site. The kiosks are not monitored by poll workers at all time and may allow voting over a period of several days or weeks. Once the voting period for a given election is up, the associated tallying site publishes the result of that election.
- **"Remote" e-voting system:** this type of systems allows voters to cast their votes from any computer or digital device connected to the Internet or to a private network; typically from home or at work. Devices such as personal digital assistance, mobile phones and even game machines may access these systems.

Several studies exist on using computer technologies in support of political remote e-voting systems. Most of these studies have focused on assessing the vulnerability of this type of systems to vote-tampering and illegal access [4]. Other studies have concentrated on developing techniques to insure the security, privacy and integrity of this type of voting systems [5].

The use of remote e-voting systems in supporting of non-political elections have been investigated by many projects [6,7]. In Ref. [8], the author develops a scheme based on the Unified Modeling Language (**UML**) to specify the requirements for mobile e-voting systems. Little work has been reported on the integration of different technology components and protocols to implement this type of systems. This paper concentrates on this aspect of the remote e-voting systems taking the implementation of an e-voting system for a university setting as a case study.

2.2. Relevance to computer standards and interfaces

This paper examines the notion of remote e-voting, and investigates the software components, standards and technologies needed to support its implementation. It also details the requirements, design and implementation of an e-voting system suitable for a university setting where students can elect their governing counsel by casting their votes anytime, anywhere and using fixed and mobile electronic devices including personal computers, personal digital assistants, and smart and regular phones.

The work presented in this paper is relevant, in many aspects, to the general area of computer standards and interfaces. First, this paper discusses and make use of Wireless and Internet communication protocols/standards including the Wireless Application Protocol (**WAP**) [9,10], Short Message Service (**SMS**) [11,12] and HyperText Transport Protocol (**HTTP**) [32]. These standards enable different electronic devices to connect to computer systems through wired, wireless and **GSM** networks.

Second, this paper uses programming standards such as the Extensible HyperText Markup Language (**XHTML**) [29], Wireless Markup Language (**WML**) [33], Extensible Markup Language (**XML**) [13], and the Extensible Style Language Transformation (**XSLT**) [14] style sheets, to represent data content and data form. These standards are instrumental in developing modern interfaces customized to fit the need of the different types of users and connecting devices. It is important to note here that the study of these standards and their use in developing modern electronic systems and interfaces are of great interest to a wide segment of "Computer Standards and Interfaces Journal" readers.

3. Requirements for the e-voting system

Prior to e-voting system design, a comprehensive and detailed set of requirements must be developed. These

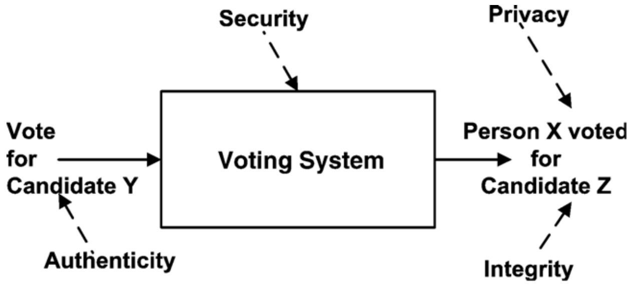


Fig. 1. A generic set of requirements.

requirements include three main types, namely, generic, system-specific and election-specific requirements. Samples of these types are presented next.

3.1. Generic requirements

The generic requirements are those requirements that apply to any voting system [8]. These requirements, as shown in Fig. 1, include:

1. **Authenticity**: only eligible voters can cast their votes;
2. **Integrity/accuracy**: once a voter cast a vote, no alternation to this vote is permitted. Moreover, all valid votes must be counted, whereas all invalid ones must not be counted;
3. **Privacy**: after casting a vote, no one should be able to link the voter to this vote;
4. **Security**: throughout the voting process, a vote can't be tampered with or viewed by anyone;
5. **Democracy**: all eligible voters must be able to vote, one person–one vote and no one can vote more than once or vote for others.

3.2. System-specific requirements

The system-specific requirements, on the other hand, are those requirements that are specific to on-line electronic-voting systems. The system-specific requirements include:

1. **Multi-user**: a number of voters can vote simultaneously;
2. **Multi-elections**: a number of elections can be running simultaneously;
3. **Accessibility**: the system can be accessed by voters at any time, from any-location using the Internet/Intranet, **PDA**s, and/or mobile phones;
4. **Availability**: the system must have high-availability during an election campaign.

3.3. Election-specific requirements

The election-specific requirements are those requirements that are specific to a given election. For example, the election-specific requirements for student council election include the following items:

1. A voter must be registered as a full time student;
2. A candidate must be registered as a full time student;
3. A candidate must have completed at least two semesters at the University and must maintain a **GPA** of at least 2.5 (out of 4.0) at the time of nomination.

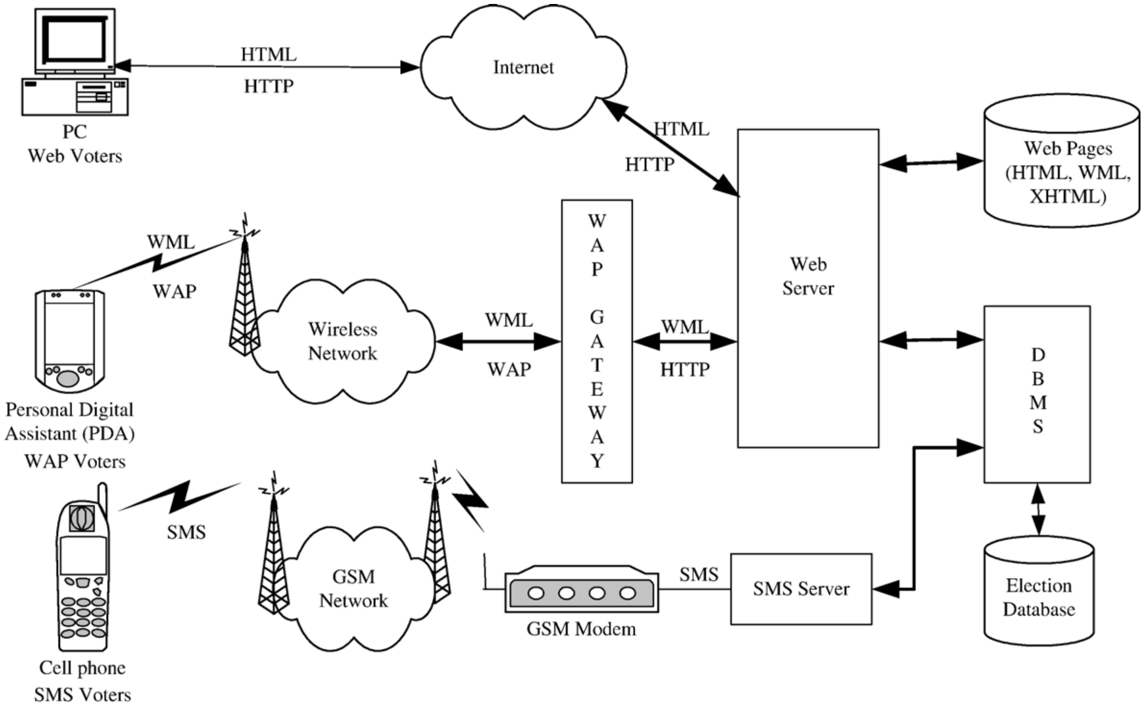


Fig. 2. The organization of a generic remote e-voting system.

Table 1
Voter types and corresponding protocols

Voter type	Access device	Presentation engine	Communication on protocol	User interface specification
Web voters	**PC**s, laptops	Web browser	**HTTP**	**HTML**, **XHTML**
WAP voters	**PDA**s, smart phones	Micro browser	**WAP**	**WML**
SMS Voters	Regular cell phones	Text editor	**SMS**	none

4. A candidate must not serve more than two consecutive years.
5. A candidate is also a voter.
6. An election winner must be one of the candidates for that election.

4. e-voting system organization

A generic e-voting system, as shown in Fig. 2, consists of a number of components, each of which will be presented next.

4.1. The election database and the database management system

The election database stores the data elements representing elections, candidates and voters as well as the schema that describes these data elements and the associated constrains. These constrains implement some of the election-specific requirements presented above. The election database is managed by a modern database management system (**DBMS**) [15] such as Oracle [16], Sybase [17] or **MS-SQL** [18].

4.2. The web server and the associated web pages

The web server connects, as shown in Fig. 2, the e-voting system with the Web and **WAP** [9,10] voters. In addition, it stores the different web pages and the code/intelligence required to interact with the user as well as the database system. The web pages are of two types, namely, static and dynamic. The data content of a static web page does not change throughout its existence, whereas, the content of a dynamic page changes based on the client input. To create dynamic web pages, a number of technologies can be used, namely, Common Gateway Interface (**CGI**) scripting [19], Personal Home Page Tools (**PHP**) [20], Active Server Pages (**ASP**) [21], Java Server Pages (**JSP**) [22] and Java Servlets [23].

4.3. The **SMS** server

The **SMS** server, as shown in Fig. 2, interacts with voters that use their regular mobile telephone sets and the **SMS** (Short Message Service) [11,12] messaging service to access the e-voting system. At the lowest level, the **SMS** server interfaces a number/bank of **GSM** (Global System for Mobile Communications) [24–26] modems that receive voters' **SMS** messages through an **SMS** service provider (**GSM** operator). Once an **SMS** message is received by a modem, the **SMS** server grabs this message, performs, on its content, the required checks and then respond to the voter by either placing a vote in the database, rendering a service to the voter or informing the voter about the occurrence of an error (e.g. invalid user name and/or password or multiple voting). After the voting campaign is completed, the **SMS** server sends all registered mobile voters (whether they have actually cast a vote or not) a message containing the election results.

4.4. The access devices

Different devices, as shown in Table 1 and Fig. 2, are used to access an e-voting system, namely, regular **PC**s or laptops, **WAP**-enabled hand-held devices (Personal Digital Assistant (**PDA**), smart phone) and/or regular cell phones. The interface technologies and protocols available for each type of these devices are outlined next.

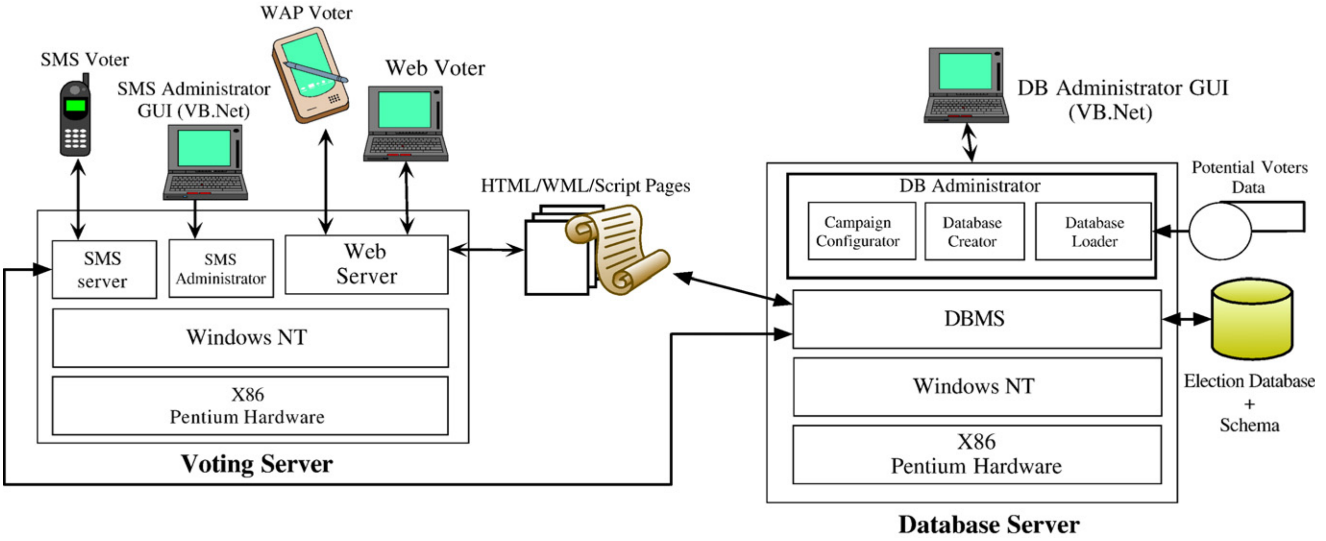


Fig. 3. The components of the implemented e-voting System.

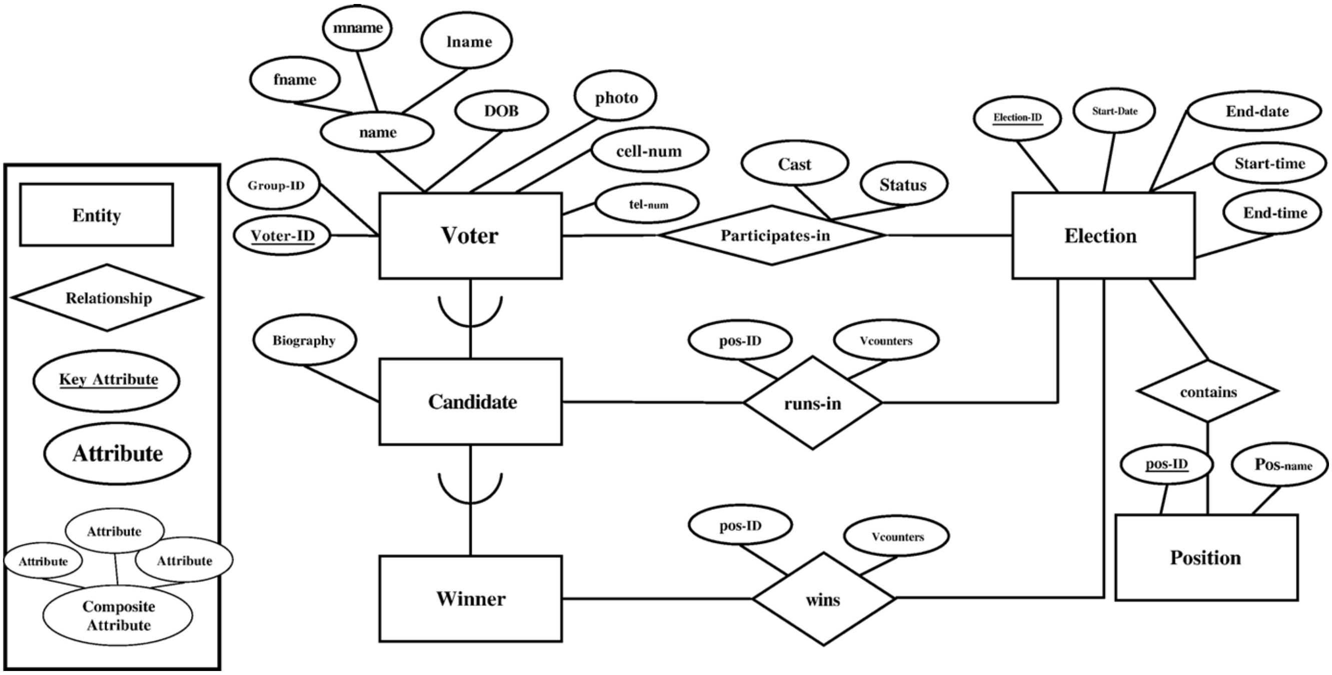


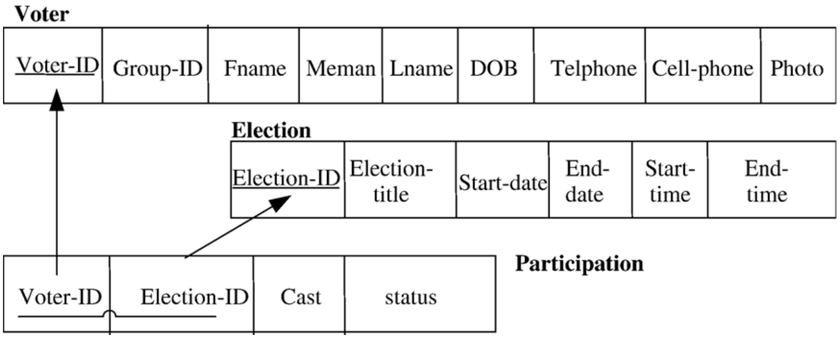
Fig. 4. The extended entity-relationship diagram for the election database.

4.4.1. Personal/laptop computers

Personal/Laptop computers, as shown in Fig. 2, connect to the e-voting system through the wired Internet. They are characterized by huge memory resources, processing power, wide communication channels, large display area and high-resolution display. **PC**/Laptop computers run full-fledged Web browsers such as the Internet Explorer or Navigator. These browsers are capable of displaying a variety of data types, including text, voice, pictures and video. This display is specified by powerful **GUI** (Graphical User Interface) languages such as the HyperText Markup Language (**HTML** 4.0) [27,28] and its successor, the Extensible HyperText Markup Language (**XHTML**) [29]. These languages are also enhanced by client scripting languages such as JavaScript [30] and VBScript [31]. To exchange information over the wired Internet, the web browser and the web server make use of the standard Internet communication Protocol, namely, HyperText Transport Protocol (**HTTP**) [32].

4.4.2. **WAP**-enabled handheld devices

WAP-enabled handheld devices, as shown in Fig. 2, connect to the e-voting system through a wireless network. Because of the network's limited bandwidth, these devices utilize the Wireless Application Protocol (**WAP**) [9,10] framework and, as shown in Fig. 2, a gateway to connect to the Web server. The role of the gateway is to translate **WAP** into the standard Internet communication protocol **HTTP** [32] and vice versa. In addition, **WAP**-enabled devices are characterized by smaller memory sizes and limited processing power and therefore they run a limited version of the standard Internet browser, referred to as **WAP** micro-browser. This browser utilizes the Wireless Markup Language (**WML**) [33] as a specification for user interfaces. Version 1.0 of **WML** is based on the Extensible Markup Language (**XML**) [13] and therefore has no resemblance to the classical **HTML** language. However, Version 2 of **WML** is based on the Extensible **HTML** (**XHTML**) [29] markup language. As a matter of fact, **WML** is a small subset of



Voter

Voter-ID	Group-ID	Fname	Meman	Lname	DOB	Telphone	Cell-phone	Photo

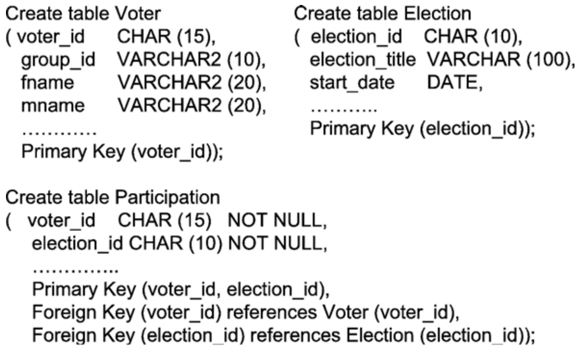
Election

Election-ID	Election-title	Start-date	End-date	Start-time	End-time

Participation

Voter-ID	Election-ID	Cast	status

Fig. 5. Part of relational schema for the election database. Arrow refers to referential integrity constrains.



```
Create table Voter                 Create table Election
( voter_id     CHAR (15),          ( election_id    CHAR (10),
  group_id     VARCHAR2 (10),        election_title VARCHAR (100),
  fname        VARCHAR2 (20),        start_date     DATE,
  mname        VARCHAR2 (20),        ...........
  ............                       Primary Key (election_id));
  Primary Key (voter_id));

Create table Participation
(  voter_id    CHAR (15)  NOT NULL,
   election_id CHAR (10) NOT NULL,
   .............
   Primary Key (voter_id, election_id),
   Foreign Key (voter_id) references Voter (voter_id),
   Foreign Key (election_id) references Election (election_id));
```

Fig. 6. Part of database schema for the election database.

XHTML that requires less processing power and therefore it is more suitable for the handheld devices. **WML** 2.0 can support tables and simple scripting through **WML**Script [34].

4.4.3. Regular cell phones

Regular cell phones, as shown in Table 1 and Fig. 2, utilize Short Message Service (**SMS**) [11,12] technology to connect users to the e-voting system. **SMS** is very popular among users because of its low cost and ease to use. The low cost stems from two factors, namely, the low cost of sending and/or receiving a message and the low cost of the mobile phone supporting this service since all that is needed is a simple textual editor for message composition and display. One disadvantage of **SMS** is its difficulty to construct a highly interactive dialog between the mobile device and the e-voting system.

5. e-voting system implementation

Following the requirements of Section 3 and the architecture of Section 4, a prototype for an e-voting system has been implemented. The developed prototype, as shown in Fig. 3, uses two hardware servers, the database and the voting ones. The functionality and software components developed for these servers are presented next.

5.1. The database server

The database server, as shown in Fig. 3, runs a modern Database Management System (**DBMS**) [15]. In addition, it supports a number of software tools, the implementation of which is presented below.

5.1.1. The election database implementation

The design and implementation of the election database starts with analyzing the election process and its data requirements and as a result, constructing the corresponding Extended Entity-relationship Diagram (**EERD**) [15]. A simplified **EERD** for the election database is presented in Fig. 4. In this diagram, the election data is organized as a set of entities (Voter, Election, Candidate, Winner) interlinked by a number of relationships. The Voter entity and its attributes, for example, capture the data associated with voters (their **ID**s, names, Date of Birth (**DOB**)… etc.), whereas, the Election entity capture the data associated with elections (election-IDs, tiles, start-date, end-date… etc.). A relationship, on the other hand, aggregates two or more entities together and associates a meaning to that aggregation. For example, the relationship *participates-in* associates a voter with a specific election.

In Fig. 4, the Candidate and Voter entities are interlinked by a specialization relationship since a candidate is indeed a voter with some extra attributes. The design process proceeds by transforming the **EERD** into a relational schema, an example of which is presented in Fig. 5, and then into a database schema expressed using the Structured Query Language (**SQL**), as shown in Fig. 6. The database is then created by the **DBMS** according to this schema. Once created, the database will be ready to receive the election data. The design and implementation of the Election database incorporates some of the Election-specific requirements as constraints. For example, election-specific requirements 5 and 6 of Section 3.3 are implemented, as shown in Fig. 4, by constructing specialization relationships between Voter, Candidate and Winner and therefore insuring that a person can't be a Winner unless he/she is a candidate and a voter.

5.1.2. The implementation of the database administrator

The database administrator tool assists the system administrator in managing the election database. The Administrator tool is a **GUI**-based tool implemented using Visual Basic (**VB**.Net) [35] framework and connects to the database using Microsoft ActiveX Data Object (**ADO**.NET) [36] technology. **ADO** provides a uniform way for a **VB** program to connect to a

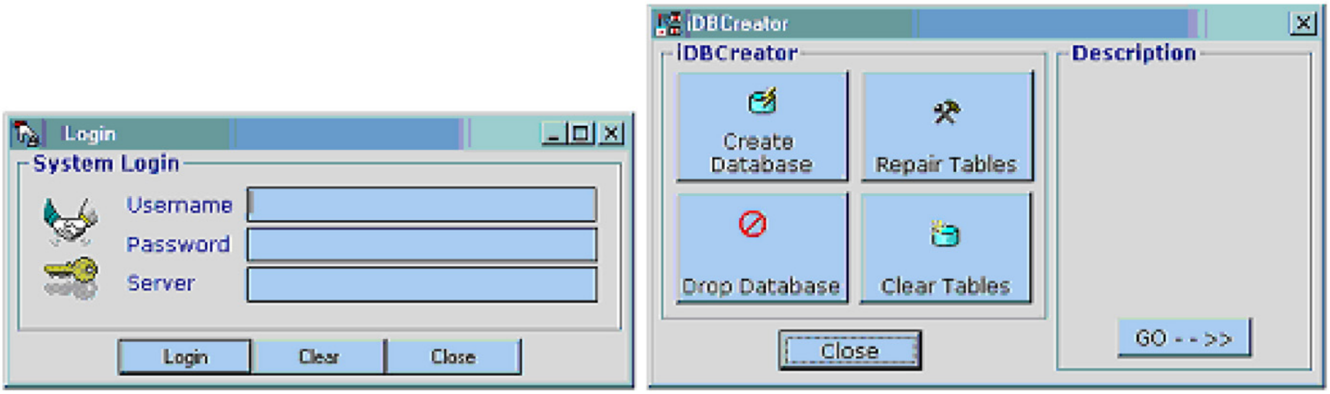
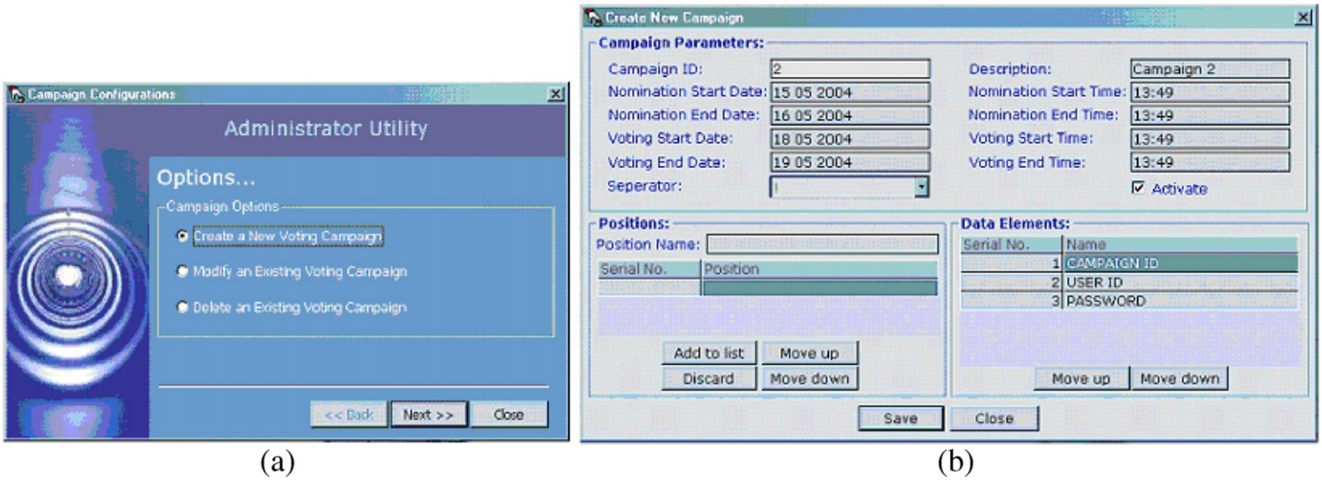


Fig. 7. The election database creator tool.


(a)

Campaign Configurations

Administrator Utility

Options...

Campaign Options

● Create a New Voting Campaign

○ Modify an Existing Voting Campaign

○ Delete an Existing Voting Campaign

(b)

Create New Campaign

Campaign Parameters:

Campaign ID:	2	Description:	Campaign 2
Nomination Start Date:	15 05 2004	Nomination Start Time:	13:49
Nomination End Date:	16 05 2004	Nomination End Time:	13:49
Voting Start Date:	18 05 2004	Voting Start Time:	13:49
Voting End Date:	19 05 2004	Voting End Time:	13:49
Seperator:	|	☑ Activate	

Positions:

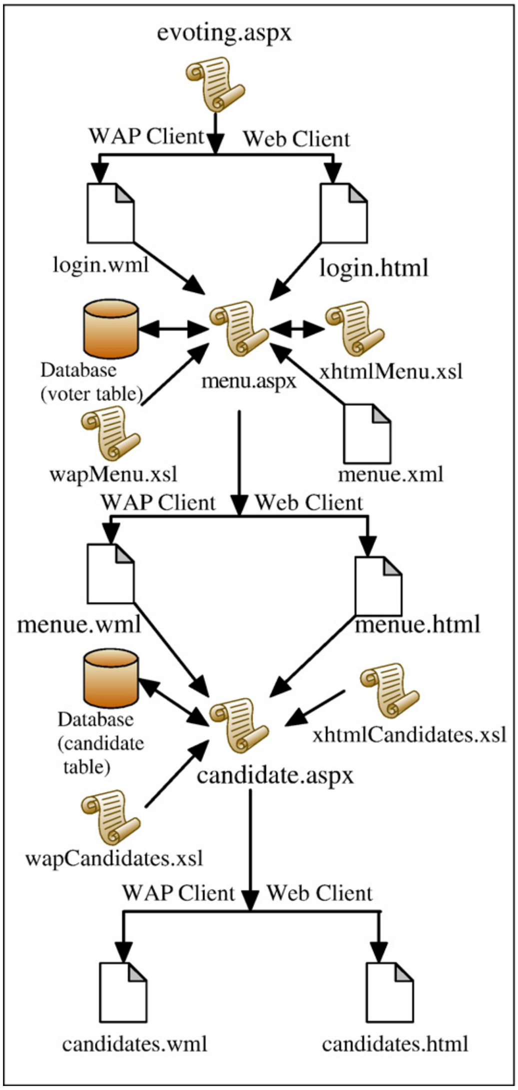
Position Name:

Serial No.	Position

Data Elements:

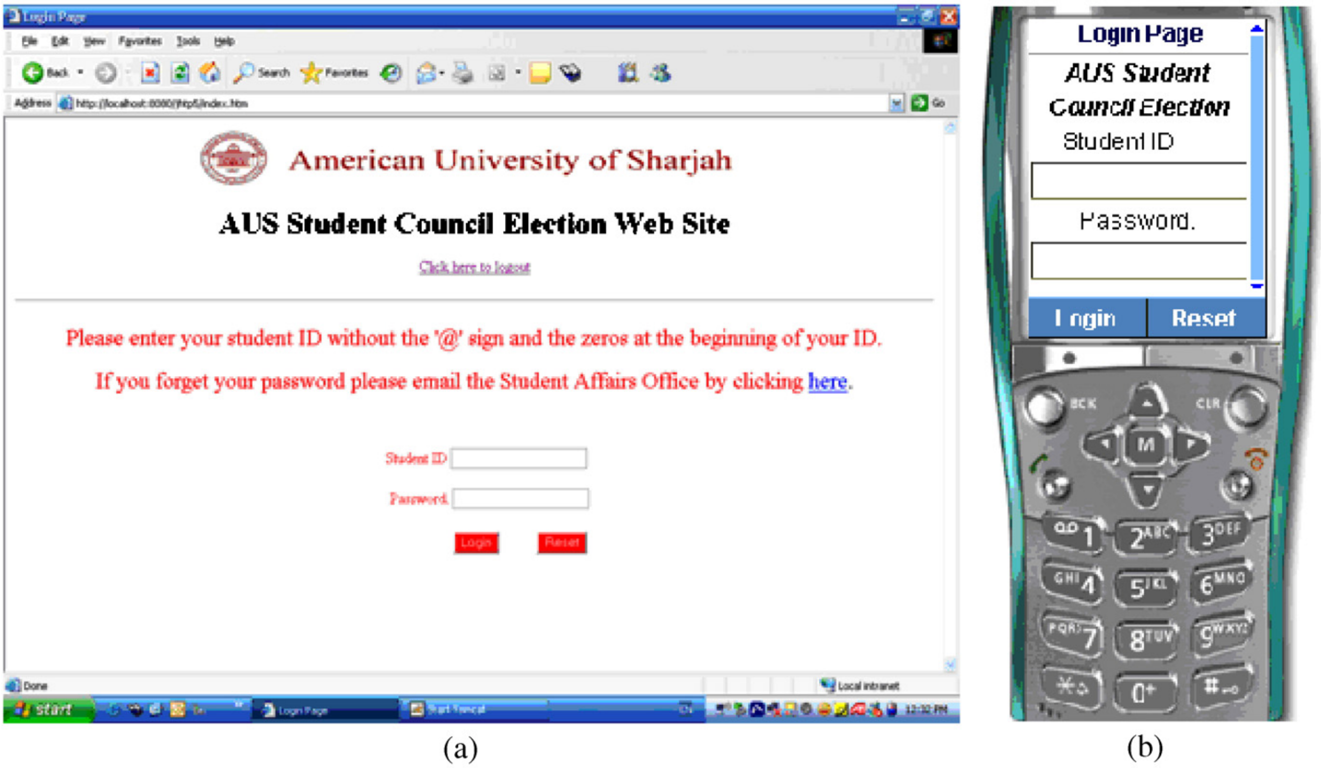
Serial No.	Name
1	CAMPAIGN ID
2	USER ID
3	PASSWORD

Fig. 8. The campaign/election configurator.



Page	Function
evoting.aspx	This is the staring page of the eVoting application. It is an Active Server Pages (ASP) with VBScript document that sends the client login.html or login.wml depending on the connecting client type.
login.html, login.wml	An XHTML or WML page that collects the user's name and password for authentication.
menu.aspx	An Active Server Pages (ASP) with VBScript document that receives user name and password and communicates with the Election database for authentication.
menu.xml	Generic Extensible Markup Language (XML) representation of the menu page/document.
xhtmlMenu.xsl	An Extensible Style Language Transformation (XSLT) document that transforms the menu XML document into an XHTML one suitable for display by a web browser.
wapMenu.xsl	An XSLT document that transforms the menu XML document into a WML one suitable for display by a WAP micro-browser.
menu.html	Dynamically generated XHTML page that displays, on a web browser, the different functionalities available for a voter.
menu.wml	Dynamically generated WML page that displays, on a WAP micro-browser, the different functionalities available to a voter.
candidates.apx	An ASP page that generates the list of candidates in a given election.
xhtmlCandidates.xsl	An XSLT document that transforms the generic XML representation of an election's candidates list into an XHTML one suitable for display by a web browser.
wapCandiates.xsl	An XSLT document that transforms the generic XML representation of an election's candidates list into a WML one suitable for display by a WAP browser.
candidates.html	Dynamically generated XHTML page that displays, on a web browser, the list of candidates in an election.
candidates.wml	Dynamically generated WML page that displays, on a WAP micro-browser, the list of candidates in an election.

Fig. 9. The server-side view of a portion of the web map.



American University of Sharjah

AUS Student Council Election Web Site

Click here to logout

Please enter your student ID without the '@' sign and the zeros at the beginning of your ID.

If you forget your password please email the Student Affairs Office by clicking here.

Student ID

Password

Login Reset

(a)

Login Page

AUS Student Council Election

Student ID

Password.

Login Reset

(b)

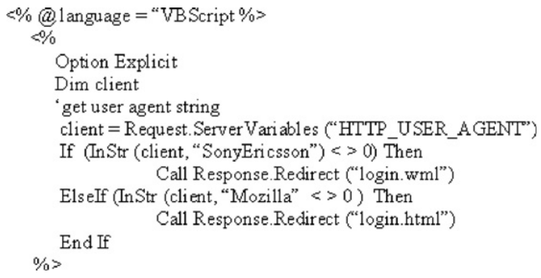
Fig. 10. Rendering login page by (a) an Internal browser, (b) a Wap micro-browser.

variety of database management systems without having to use **DBMS**-specific features. The different components of the administrator utility are presented next.

- **The database creator**: the database creator tool, as shown in Fig. 7, creates the election database, according to the associated schema. In addition, this tool is used to clear out the database tables and/or dropping the database.
- **The database loader**: the database loader tool, as shown in Fig. 3, enables the system administrator to load, from a file, qualified voters into the Voter table. To do so, the database loader implements some of the election-specific requirements as constraints and apply these constraints to the file to select those individuals that are eligible for voting and, then, load these individuals into the Voter Table. Users may also register for voting through the Internet (Web and **WAP** users) and/or **SMS** messages.
- **The campaign configurator**: this tool, as shown in Figs. 3 and 8a, is used to create, delete and/or modify election campaigns. The creation of a campaign/election is carried out using the form presented in Fig. 8b, whereas, other forms similar to the one in Fig. 8a have been implemented. Using these forms, one may add a new candidate to an election and may add a new voter to the system.

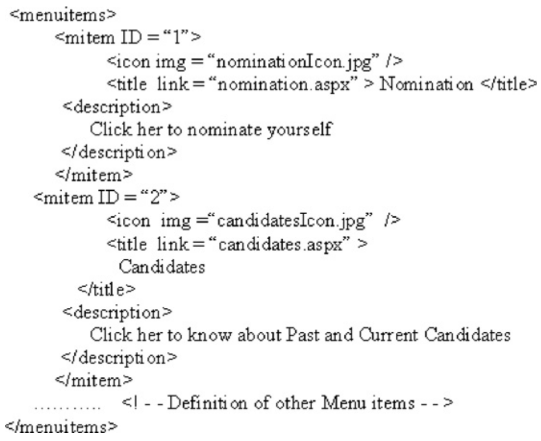
5.2. The voting server

The voting server, as shown in Fig. 3, runs a modern Internet Web Server, the Internet Information Server (IIS) for example, in support of the Web and **WAP** voters. In addition, the voting server supports a number of software components, the implementation of which is presented next.



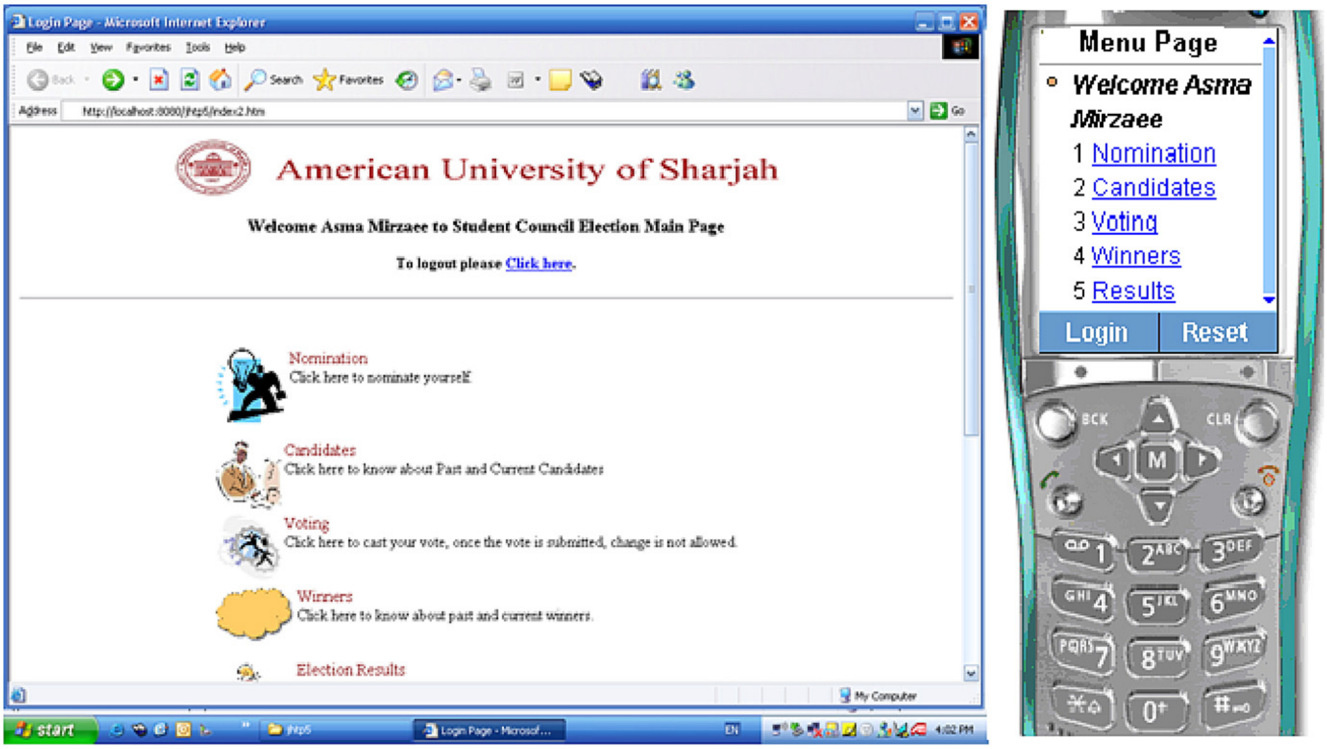
```
<% @language = “VBScript %>
  <%
    Option Explicit
    Dim client
    ‘get user agent string
     client = Request.ServerVariables (“HTTP_USER_AGENT”)
     If  (InStr (client, “SonyEricsson”) < > 0) Then
                 Call Response.Redirect (“login.wml”)
     ElseIf (InStr (client, “Mozilla”  < > 0 )  Then
                 Call Response.Redirect (“login.html”)
     End If
  %>
```

Fig. 11. A **VB**Script segment.



```
<menuitems>
      <mitem ID = “1”>
             <icon img = “nominationIcon.jpg” />
             <title  link = “nomination.aspx” > Nomination </title>
       <description>
           Click her to nominate yourself
       </description>
      </mitem>
   <mitem ID = “2”>
             <icon  img =“candidatesIcon.jpg”  />
             <title  link = “candidates.aspx”  >
               Candidates
         </title>
       <description>
           Click her to know about Past and Current Candidates
       </description>
      </mitem>
   ...........     <! - - Definition of other Menu items - - >
</menuitems>
```

Fig. 12. An **XML** description of the menu page.



Welcome Asma Mirzaee to Student Council Election Main Page

To logout please Click here.

Click here to nominate yourself.

Click here to know about Past and Current Candidates

Click here to cast your vote, once the vote is submitted, change is not allowed.

Click here to know about past and current winners.

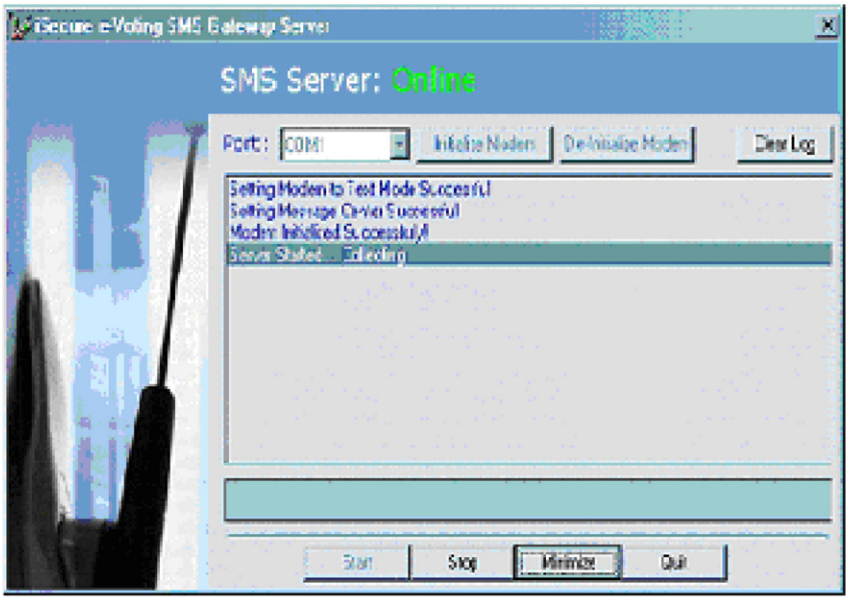
Welcome Asma Mirzaee

(a) (b)

Fig. 13. Rendering the menu page by (a) an Internet browser, (b) a Wap micro-browser.

5.2.1. The implementation of the voter interface

The implementation of the voter interface adopts the philosophy “author once, publish on any device”. To achieve this goal, we have utilized the power of the Extensible Markup Language (**XML**) [13] to represent data content and the power of Extensible Style Language Transform (**XSLT**) [14] style sheets to present this content to different voter devices. In presenting the voter interface, we distinguish between the client-side view, defined as the sequence of web pages rendered by an Internet browser (web voters) or micro-browser (**WAP** voters), and the server-side view defined as the set of web pages that generates the client-side ones. The web pages within the client-side view are written in **XHTML** [22] and JavaScript (web clients) [29] and **WML** [33] and **WML**Script (**WAP**



Setting Modem to Text Mode Succesful
Setting Message Center Successful
Modem Initialized Successfully
Server Started... Listening

Fig. 14. The **SMS** Server Administrator tool.

clients) [34]. Theses pages include the "login page" which represents the home page for the e-voting system, the "menu page" which contains buttons for the different voting functions, the "nomination page" which enables a user to nominate self or a colleague, the "candidates page", the "voting page" … etc. Some of these pages, such as the "login page", are stored as is in the Web server, whereas, some others, such as the "candidates page", are generated using server-side script documents. The server page that is rendered as is by the client is referred to as static, whereas, the pages that is modified/generated by a server-side script before it is rendered by the client is referred to as dynamic.

Fig. 9 presents part of the server-side web pages/map as well as their definition, functionality and content. At the top of this map is the active server pages (**ASP**) [13] document "evoting.aspx" with Visual Basic Script (**VB**Script) [35] as the scripting language. When referenced, as shown in Fig. 9, evoting.aspx identifies the type of connecting client and redirects the reference to the appropriate login page. If the client type is a web-browser, Internet Explorer or Internet Navigator for example, then the client is redirected to login.html, which gets rendered as shown in Fig. 10a, whereas, if the client type is a **WAP**-based micro-browser, then, the client is redirected to login.wml which gets rendered as shown in Fig. 10b.

The **VB**Script, presented in Fig. 11, shows a segment of evoting.aspx. When a client connects to a web server, the former stores an identifying string of characters into the server's **HTTP**_USER_AGENT environment variable. The **VB**Script segment, presented below, then searches this string for a keyword that identifies the type of the connecting client. The appropriate login page then gets identified, based on the search outcome, and sent to the client for display.

The rendered version of login.html or login.wml, as shown in Fig. 10, contains two input fields, the student **ID** and password. Once the user types his/her data into these fields and presses the Login button, the client directs the entered data to the server **ASP** document menu.aspx. As a result, menu.aspx creates and sets up an ActiveX Data Object (**ADODB**) and connects to the Voter table of the Election database. Through this object, menu.aspx retrieves the data needed to authenticate users. If the authentication is successful, then, menu.aspx loads a generic **XML** representation of the menu page, a fragment of which is shown in Fig. 12, applies the appropriate **XSLT** sheet (xhtmlMenu.xsl or wmlMenu.xsl depending on the type of the connecting client) to generate menu.html, an **XHTML** document, or menu.wml, a **WML** document. The resulting document is then sent to the connecting client for display. Fig. 13 presents menu.html and menu.wml as rendered by the Web and **WAP** browsers, respectively.

Pressing the Candidates link on the menu page of Fig. 13, transfers the control, as shown in Fig. 9, to another **ASP** document, namely, candidates.aspx which first creates and sets up an **ADO** object, through which it connects to the Candidates database table. The processing continues by retrieving the list of candidates for the specified election and generating an **XML** document that represents the candidates' retrieved list. candidates.aspx continues by applying the appropriate **XSLT** sheet (xhtmlCandidates.xsl or wmlCandidates.xsl depending on the type of the connecting client) to generate candidates.html, an **XHTML** document, or candidates.wml, a **WML** document. The resulting document is then sent to the connecting client for display.

5.2.2. *The* **SMS** *server implementation*

The **SMS** server, as shown in Fig. 3, is a software process/service running on the voting server. It monitors the **GSM** modems, process the received **SMS** messages and interact with the database server accordingly. Different message types are processed by the **SMS** server, namely, registration for taking part in an election, self-nomination, others-nomination, voting in an election and requesting information. To insure security, each message sent to the voting **SMS** Server, with the exception of information requests, is accompanied by user name and password. The implementation of the **SMS** process utilizes Visual Basic as a programming language. It also utilizes **ADO**. Net technology to connect to the Election database. The **SMS** Server is administered by an implemented **GUI**-based tool, presented in Fig. 14. This tool is used to configure, run and stop the **SMS** Server as well as initializing the **GSM** modem bank.

6. Conclusion

The requirements, design and implementation of an electronic voting system have been presented. The system can be accessed anytime, anywhere and using different devices including **PC**s, through fixed-wire Internet/Intranet, **WAP**-enabled mobile devices such as Personal Digital Assistants (**PDA**s) and smart phones, and **SMS**-based regular phones. For elections with low to medium security requirements, such as those elections found in a University setting, any or all of these devices and techniques can be used by voters to cast their votes. To insure security in this case, the voter needs to supply his/her user **ID** and password only. However, for highly secure elections, such as political ones, voters need to access the e-voting system through secure channels including the use of secure client devices located at secure polling locations and connected to the e-voting system through secure Intranets/private networks.

To avoid web content replication for each of the client devices, the implemented system separates the data content from its presentation form. To achieve such a separation, the Extensible Markup Language (**XML**) is used to represent the web data content, whereas, the Extensible Style Language Transformation (**XSLT**) style sheets are used to customize the presentation of such content on different connecting devices, thus, achieving true "author once, publish to any device" design and implementation.

Acknowledgement

The author Ghassan Qadah would like to acknowledge the support of this research by a research grant from the American University of Sharjah.

References

[1] Postnote: Online Voting. UK Parliamentary Office of Science and Technology, May 2001, Number 155, pp. 1 – 4. Online: www.parliament.uk/post/pn155.pdf.

[2] D. Dill, R. Mercuri, P. Neumann, D. Wallach, Frequently Asked Questions about DRE Voting System, Feb. 2003. Online: http://www.verifiedvoting.org/drefaq.asp.

[3] M. Shamos, Paper v. electronic voting records — an assessment, Proceedings of the 14th ACM Conference on Computers, Freedom and Privacy, May 2004.

[4] T. Kohno, A. Stubblefield, A. Rubin, D. Wallach, Analysis of an electronic voting system, Proceedings of IEEE Symposium on Security and Privacy, May 2004, pp. 27–40.

[5] A. Rubin, Security considerations for remote electronic voting over the internet, Login: The Magazine of USENIX & SAGE 26 (1) (February 2001) 20–28.

[6] S. Stuart, M. Brown, S. Draper, Using an electronic voting system in logic lectures: one practitioner's application, Journal of Computer Assisted Learning 20 (2) (April 2004) 95–102.

[7] S. Draper, M. Brown, Increasing interactivity in lectures using an electronic voting systems, Journal of Computer Assisted Learning 20 (2) (April 2004) 81–94.

[8] K. Saleh, Specifications of a mobile electronic voting system and a mobile agent platform, Proceedings of IADIS e-Society, Lisbon, Portugal, June 2003, pp. 281–287.

[9] GSM World: What is WAP? Online: http://www.gsmworld.com/technology/wap/intro.shtml.

[10] WAP Specification Releases. Online: http://www.wapforum.org/what/technical.htm.

[11] GSM World: What is SMS? Online: http://www.gsmworld.com/technology/sms/intro.shtml.

[12] SMS Forum. Online: http://www.smsforum.net.

[13] Extensible Markup Language (**XML**) 1.0. Online: http://www.w3.org/TR/REC-xml.

[14] XSL Transformations (**XSLT**) Version 1.0. Online: http://www.w3.org/TR/xslt.

[15] R. Elmasri, S.B. Navathe, *Fundamentals of Database Systems*, 4th ed., Pearson/Addison Wesley Publisher.

[16] J. Morrison, M. Morrison, Guide to Oracle 9i, Thomson Course technology, 2003.

[17] D. McGoveran, C.J. Date, *A Guide to Sybase and SQL Server*, Addison-Wesley Publisher.

[18] S. Bjeletich and G. Mable, *Microsoft SQL Server 7.0 Unleashed*, Sams Publisher.

[19] S. Gundavaram, G. Birznieks, S. Guelich, CGI Programming with Perl, 2nd ed.Oreilly & Associates, 2000.

[20] H.M. Deitel, P.J. Deitel, T.R. Nieto, K. Steinbuhler, Internet, World Wide Web How to Program, 2nd ed., Prentice Hall, 2002, pp. 1008–1055.

[21] H.M. Deitel, P.J. Deitel, T.R. Nieto, K. Steinbuhler, Internet, World Wide Web How to Program, 2nd ed., Prentice Hall, 2002, pp. 831–907.

[22] H. Bergsten, JavaServer Pages, 3rd ed.Oreilly & Associates, 2003.

[23] J. Hunter, W. Crawford, Java Servlet Programming, 1st ed.Oreilly & Associates, 1998.

[24] GSM World: http://www.gsmworld.com/technology/gsm.shtml.

[25] M. Mouly, M. Pautet, *The GSM System for Mobile Communications*, order information: http://www.telecompublishing.com/gsm-system-for-mobile.shtml.

[26] J. Scourias, Overview of the Global System for Mobile Communications. Online: http://www.shoshin.uwaterloo.ca/~jscouria/GSM/gsmreport.html.

[27] HyperText Markup Language Home Page: Online: http://www.w3.org/MarkUp/.

[28] HTML 4.01 Specifications. Online: http://www.w3c.org/TR/html401.

[29] XHTML 1.0: The Extensible HyperText Markup Language. Online: http://www.w3.org/TR/xhtml1.

[30] H.M. Deitel, P.J. Deitel, T.R. Nieto, K. Steinbuhler, Internet, World Wide Web How to Program, 2nd ed., Prentice Hall, 2002, pp. 194–434.

[31] H.M. Deitel, P.J. Deitel, T.R. Nieto, K. Steinbuhler, Internet, World Wide Web How to Program, 2nd ed., Prentice Hall, 2002, pp. 784–830.

[32] Fielding, et al. HyperText Transport Protocol - HTTP/1.1 RFC 2616, June 1999. Online: http://www.ietf.org/rfc/rfc2616.txt.

[33] H.M. Deitel, P.J. Deitel, T.R. Nieto, K. Steinbuhler, Wireless Internet, and Mobile Business — How to Program, Prentice Hall, 2002, pp. 446–489.

[34] H.M. Deitel, P.J. Deitel, T.R. Nieto, K. Steinbuhler, Wireless Internet, and Mobile Business — How To Program, Prentice Hall, 2002, pp. 492–660.

[35] R. Steven, R. Petrusha, P. Lomax, *VB.NET Language in a Nutshell*, 2nd ed., O'RELLY.

[36] B. Hamilton, ADO.NET Cookbook, O'RELLY, 2003.

Ghassan Z. Qadah obtained his BSc from the Electrical Engineering Department of Ain-shames University, Cairo, Egypt, and his MSc. and PhD. from the Electrical and Computer Engineering Department of the University of Michigan–Ann Arbor. Currently, he is an Associate Professor at the Computer Engineering Department, American University of Sharjah, P.O. Box 26666, Sharjah, United Arab Emirates. His research interest includes serial and parallel query processing algorithms, traditional and non-traditional database systems, wireless networks and mobile computing.

Rani Taha obtained his BSc from the Computer Engineering Department of the American University of Sharjah, UAE. His interest includes GUI development and mobile computing.