An Extension of QSL for E-voting Systems

Yuan Zhou, Hongbiao Gao, and Jingde Cheng(✉)

Department of Information and Computer Sciences,
Saitama University, Saitama 338-8570, Japan
{shuugen, gaohongbiao, cheng}@aise.ics.saitama-u.ac.jp

Abstract. E-voting is an electronic way to provide voting processes beginning from preparing ballots, following by authenticating voters and candidate registrations, through casting votes, and ending to tallying and declaring collected answers. Nowadays, there are many kinds of e-voting systems implemented to provide e-voting services over the Internet. However, there is no ad hoc method to cover the gap caused by difficult communications. QSL is a specification language for e-questionnaire systems that serves as a communication tool for specifying e-questionnaires and e-questionnaire systems. QSL is an ideal candidate because of similar processes between e-questionnaire and e-voting. The current version of QSL is reckoned without e-voting and e-voting systems. This paper proposes an extension of QSL for specifying e-voting and e-voting systems, and presents two cases using QSL for e-voting systems to show its effectiveness.

Keywords: QSL · Specification language · E-voting system · E-voting · XML · Security · Web service

1 Introduction

Elections, referenda and polls are critical processes for appropriate operation of a modern democracy [5]. Because of high efficiency and low cost of counting votes, and accessibility and convenience for disabled voters and the voters who live in remote place, many countries began to use electronic voting (e-voting) technology. E-voting is an electronic way to provide voting processes beginning from preparing ballots, following by authenticating voters and candidate registrations, through casting votes, and ending to tallying and declaring collected answers.

Over a decade, many kinds of e-voting systems are implemented to provide e-voting services over the Internet from any place and any computer or computerized equipment connecting to the Internet. Based on law, regulation, and policy, different requirements for different types of e-voting can thus derive different e-voting systems. When the government, party, and organization want to use the existing e-voting system to do an e-voting, they must specify e-voting at first to figure out the requirements. When the existing e-voting systems cannot satisfy the requirements what they want, they should order an e-voting system. It is also need to specify the e-voting system helping to clear what are necessary, and to make a clear explanation with the implementers who are not the specialists in election. Therefore, it is necessary to provide a specification language, which can help them to create precise and adequate specifications for various e-voting and e-voting systems.

QSL [2, 30] is a specification language for e-questionnaire systems that serves as a general-purpose communication tool for specifying various e-questionnaires and e-questionnaire systems with a standardized, consistent, and exhaustive list of requirements. It is an ideal candidate on account of its creative concept and similar processes between e-questionnaire and e-voting, generally manifested in steps of setting up, distributing, submitting, collecting, and counting. However, the current version of QSL is reckoned without e-voting and e-voting systems.

This paper proposes an extension of QSL for e-voting systems such that users can use QSL to specify various e-voting and e-voting systems. The rest of the paper is organized as followed: Sect. 2 gives introduces in QSL. Section 3 presents an extension of QSL for e-voting systems. Section 4 shows two cases of representative e-voting systems being specified by extended QSL to confirm that QSL is hopeful to specifying e-voting systems. The related work is presented in Sect. 5. Finally, some concluding remarks are given in Sect. 6.

2 QSL: A Specification Language for E-questionnaire Systems

QSL is a specification language for specifying various e-questionnaires and e-questionnaire systems [30]. QSL serves as a communication tool with a standardized, consistent, and exhaustive requirement list to provide services to both questioner and implementer that can make questioners clearly describe the requirements for an e-questionnaire, and implementers can also clearly understand what questioners need. In addition, implementers can implement the e-questionnaire system. QSL also supports communication between questioners and an e-questionnaire system to provide only one method for questioners. Meanwhile, QSL provides services for questioners with e-questionnaire data as only one format that can be reused. QSL has an ideal state that is a tool as a complier to automatically generate e-questionnaire system for questioner [2].

QSL is based on XML [27] to provide a well-formed structure. The grammar of QSL uses XML Schema [28] to describe the structure of the documents with a precision and conciseness. QSL provides specifications of exhaustive requirements by the method of combining the primitive elements of e-questionnaires and e-questionnaire systems. QSL has been extended to specify e-testing and e-testing systems [26].

3 An Extension of QSL for E-voting Systems

3.1 E-questionnaire and E-voting

In order to extend QSL for e-voting systems, we investigated 20 e-voting systems [1, 3, 4, 6, 8, 10–12, 14–23, 25, 29] and 119 requirements [24] of e-voting systems deduced from legal input and accepted by the lawyers.

We found the similarities between e-questionnaire and e-voting from 6 aspects, which are phase, e-paper, server, software, function, and participant. Firstly, both e-questionnaire and e-voting are not out of phases of setting up, distributing, submitting, collecting, and counting. Setting-up is to prepare software communicating with server and e-paper needed for an event. Distributing is to distribute e-paper to respondents. Submitting is to answer e-paper and send to submitting server, usually called voting phase in e-voting. Collecting is to collect the answers from respondents. Counting is to calculate the collected answers and get results, usually called tallying phase in e-voting. Secondly, e-questionnaire and e-ballot have the extremely similar contents used to express a choice preference, collectively called e-paper. E-paper consists of settings, questions, and options. Thirdly, both e-questionnaire and e-voting need server to store the collected results and server to provide registration services for respondents. Fourthly, according to phases, they have similar software to provide communication with corresponding servers. There are 2 kinds of software, one is client-side software to communicate with submitting server, another is counting software to communicate with counting server. Fifthly, both e-questionnaire and e-voting have similar functions based on phases. For instance, both e-questionnaire and e-voting need provide function of allowing access to the server if at least two different users are logged on in submitting phase. Lastly, both e-questionnaire and e-voting have similar roles of participants, which are sponsor who organizes and supports an event, questioner who designs an e-paper, analyst who processes the collected answers, monitor who monitors whether illegal or dishonest behavior occurs or not, and respondent who answers the e-paper, usually called voter in e-voting.

Moreover, we found the differences between e-questionnaire and e-voting mainly manifested in security because e-voting is mainly used in governmental elections for the universal, equal, free, and secret suffrage. More specifically, the points of a trustworthy secure e-voting system differing from e-questionnaire system are following aspects: authentication and anonymisation. Firstly, authentication is to ensure only eligible voters may cast vote only once before storing in the e-ballot box, and must ensure the casted votes are clearly separated from the identity of the voter. For instance, in order to ensure no ineligible voter cast a vote for changing election results, e-voting systems always use tokens as an authentication technique. Secondly, anonymisation is to prevent any link between the voter and his unencrypted vote. It concerns the communication channel encryption and seal method. In addition, thirdly, e-voting needs auditing phase to verify and ensure the equality of the number of voters and votes in e-ballot boxes. Auditing provides services of recording, monitoring, and verification of audit data to make authenticity and accuracy of voting results, for the security of e-voting. Fourthly, e-voting also needs certification server to provide services to validate the voters and poll workers we will explain below to prevent any possibility of affecting election results. Fifthly, e-voting needs an auditing software to communicate with submitting system. Besides, sixthly, e-voting needs a list of candidates, and provides a candidate nomination and candidate registration. At last, e-voting also needs voting worker to help start submitting phase, make a selection on e-paper, resume submitting phase after any kind of exception, malfunction, or breakdowns, check the system state, close submitting phase, and start counting phase, especially in parliamentary elections.

3.2 Extending QSL for E-voting Systems

According to the similarities and differences between e-questionnaire and e-voting, we propose an extension of QSL for e-voting systems based on the current QSL for e-questionnaire and e-testing systems, and we change properly the structure of it. Figure 1 illustrates QSL structure of relationship overview among e-questionnaire, e-testing, and e-voting.

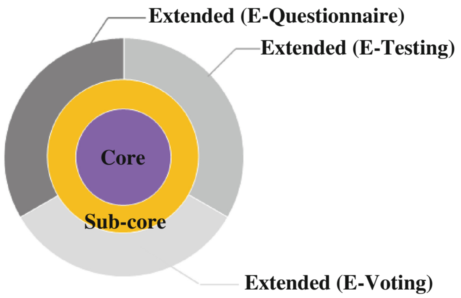

Fig. 1. Relationship overview among e-questionnaire, e-testing, and e-voting.

In order to extend and upgrade QSL easily, we design that QSL structure has 3 layers. In the innermost layer, QSL defines core elements. Specifying any e-questionnaire, e-testing, and e-voting system must specify all the core elements. The core element consists of the combinations of the elements in the middle layer. The elements in middle layer are called sub-core elements. In the outermost layer, there are 3 isolated ranges, which are for e-questionnaire, e-testing, and e-voting, respectively. The elements in this layer are called extended elements. For example, if a user wants to specify an e-voting system, he/she shall specify all the elements in the innermost layer and middle layer, and specify all or part of the elements in the outermost layer depending on the security level.

In order to well define the combinations of core elements, sub-core elements, and extended elements, we use double-digit to mark the elements. In the ten's place, 0, 1, and 2 stands for core element, sub-core element, and extended element, respectively. Table 1 shows a list of the elements with the double-digit numbers. In the one's place, 0 stands for a special mark. The elements are associated with the namespace defined using **QSL**. As the configuration of core elements, it gives a combination relationship of numbers of sub-core elements and extended elements defined only for e-voting. As well, some major elements for constructing sub-core elements are shown below.

Based on the current QSL, we exact **Security** as a core element from **System**. To well deal with the intruder's technical capabilities, from core, through sub-core, to extended elements, QSL exhibits its extensibility to easily revise elements in middle and outermost layers. We design QSL providing the phase security isolation. In other words, there are different security requirements in each phase.

In security aspect, we exact **Authentication** and **Anonymisation** as two important elements in the outermost layer. **Authentication** has 3 child elements, which are

Table 1. Core elements, sub-core elements, and extended elements for e-voting.

Group	No.	Element	Configuration
Core	00	QSL	–
	01	Security	Phase, Server, Software, Participant
	02	System	Phase, Function, Server, Software
	03	EPaper	Phase, Paper, Participant
	04	Data	Phase, Paper, Participant
Sub-core	11	Phase	SettingUp, Distributing, Collecting, Submitting, etc.
	12	Paper	Arrangement, Text, Media, Question, Option, etc.
	13	Function	Func-Import, Func-Export, Func-Distribute, etc.
	14	Server	RAServer, PSServer
	15	Software	CSSoftware, CTSofteware
	16	Participant	Sponsor, Questioner, Analyst, Monitor, Respondent
Extended (E-V)	21	Authentication	Secret, Token, Biometrics
	22	Anonymisation	Seal, Channel
	23	Auditing	–
	24	CAServer	ID, Link
	25	ATSoftware	Version, ID, Link, Solution
	26	Candidate	ID, Name, Affiliation, Proposer
	27	Admin	–

Secret, **Token**, and **Biometrics** in a choice relationship. Each child element is designed as string type and derived by restriction with enumeration values. An example is shown below. To maximize the security for difficult to fake the card, the most popular method is the combination of knowledge of a secret and ownership of a token. *TAN* is a unique code of letters and digits send by a secure post to eligible voters in election setup phase. *ID Card* is pre-existing election authentication card.

```
<Authentication>
   <Secret>TAN</Secret>
   <Token>ID Card</Token>
</Authentication>
```

As to **Anonymisation**, it has 2 child elements, which are **Seal** and **Channel**. **Seal** has an attribute to specify the seal method and a content to specify what is sealed. To ensure secure communication channel, the value of the element contains “Internet Link”, “Phone”, “E-mail”, and so on. **Channel** has an optional attribute named *ID*. The optional attribute *Method* belongs to **Anonymisation** to specify the method containing “BlindSignature”, “SeparationOfDuty”, “HardwareSecurityModel”, etc.

```
<Anonymisation>
   <Seal Method="Signature">Vote</Seal>
   <Channel>Internet Link</Channel>
</Anonymisation>
```

In order to specify different servers in an event for their different functions and authority, in essence for security, **Server** has 2 groups, one is called “ServerGroup”, another is called “SeparateServerGroup”. The first group is used in a situation that all the phases are executed on a server, which has 2 child elements named **ID** and **Link**. It is used to deal with the communication links between software and servers. Alternatively, the second group is used to separate into 3 servers according to the phases. In this group, there are 3 child elements, which are **RServer** (Registration Server) to let voter register, **CAServer** (Certification Server) to confirm whether the voter is eligible or not, and **PSServer** (Paper Storage Server) to store the e-votes in e-ballot boxes. Each child element has a child group named “ServerGroup” in order to connect with corresponding servers. For example, in below situation, the registration server named “Re1” links to a software named “Soft1”.

```
<RServer>
   <ID>Re1</ID>
   <Link Ref="SoftID">Soft1</Link>
</RServer>
```

Corresponding to server and system, **Software** has 2 groups, one is called “SofrwareGroup”, another is called “SeparateSoftwareGroup”. The first group is used in a situation that all the phases are executed using software, which has an optional child element named **Version**, and 3 necessary child elements named **ID**, **Link**, and **Solution** in sequence. The second group is used to separate into 3 kinds of software according to the phases for security, especially in auditing phase. In this group, there are 3 child elements, which are **CSSoftware** (Client-side Software) to communicate with submitting system, **CTSoftware** (Count Software) to communicate with counting system, and **ATSoftware** (Auditing Software) to communicate with auditing system. Each child element has a child group named “SoftwareGroup” in order to connect with corresponding systems. For example, the software named “Soft1” links to an e-voting system named “Sys1”, and uses web browser to perform an e-voting.

```
<Software>
   <ID>Soft1</ID>
   <Link Ref="SysID">Sys1</Link>
   <Solution Type="Web Browser"/>
</Software>
```

We add **Auditing** as the last phase, in the sequence of child elements of **Phase**. It can combine with **Server** and **Software**. In addition, we add **Admin** as a role of poll worker, in the sequence of child elements of **Participant**. Because poll worker has different duties in an event to prevent bribe, **Admin** has an attribute *Role* to distinguish each duty related to authority. Besides, we add **Candidate** and its child element, which are **ID** to identifier, **Name** as necessary information, **Affiliation** as optional information, and **Proposer** related with nomination. The candidate information is used as an option of the question. The nomination is similar to a questionnaire proposed by a group of respectable people, is used to get candidate list as question options. It can combine with **RAServer** to specify candidate registration.

4 The Cases of Using QSL for E-voting Systems

We used extended QSL to specify the Estonian system, which was used to hold a federal election in Estonia in 2007 [7]. In addition, we also specify POLYAS system [13], a famous voting system founded in 1996 has been used to cast more than one million vote a year for its elections. As the results, the extend QSL can specifying the whole requirements, especially all the security requirements. In other words, the elements in QSL structure for e-voting systems can be used to specify these 2 systems. As examples, two representative security specification snippets for submitting phase of Estonian System and setting up phase of POLYAS System by QSL.

Estonian System uses a combination of possession-based and secret-based authentication method. The voter used the first secret key, an ID card, to identify the authority of casting a vote and the second secret key to sign his encrypted vote. Voter cannot change his vote after casting a vote. In submitting phase, the voter chooses and the system shows his choice to let voter to verify or change his choice again before submitting vote to the e-ballot box. When he submits his confirmed vote, voter uses his second secret key "Signature" and a blind signature scheme to encrypt the vote. In this snippet, it specifies the authentication method named "Signature" the voter has, a secure Internet link, and a sealed and encrypted anonymous method. In addition, only an eligible voter can cast a vote. If any kind of exceptions, malfunction, or breakdown occurs, the certified poll workers can resume this phase. The certified monitor can monitor the phase. Both worker and monitor cannot authority to see the votes.

```
<Submitting>
   <Authentication>
       <Biometrics>Signature</Biometrics>
   </Authentication>
   <Anonymisation>
        <Seal Method="Signature">Vote</Seal>
        <Channel>Internet Link</Channel>
   </Anonymisation>
   <Authority>
      <Participant>
        <Monitor Role="Certified" Situation="Phase"/>
        <Admin Role="Certified" Situation="Phase"/>
        <Respondent Role="Certified"
Situation="Ballot"/>
      </Participant>
   </Authority>
</Submitting>
```

POLYAS System uses the secret-based authentication method named "TAN", and the separation of duty approach to anonymise voter's identity among multiple servers. After logging on client-side voting software, voter uses his secret key through the first directed SSL connection from the software and registration server. The server checks whether the requesting voter is eligible or not. In order to ensure one voter one vote, the ID code is sent to the vote storage server through the second directed SSL connection, which checks whether the voter has already cast a voter. If this voter has not vote, the registration server generate a ballot belongs to this particular voter.

```
<SettingUp>
  <Authentication>
    <Secret>TAN</Secret>
  </Authentication>
  <Anonymisation Method="SeparationOfDuty">
     <Channel ID="SSL1">Internet Link</Channel>
     <Channel ID="SSL2">Internet Link</Channel>
  </Anonymisation>
  <Authority>
    <Participant>
      <Respondent Role="Certified"/>
    </Participant>
  </Authority>
</SettingUp>
```

5 Related Work

EML is a standard for the structured interchange of data among hardware, software, and service providers who engage in any aspect of providing election or voter services to public or private organizations [9]. EML is a uniform and reliable method to allow systems supporting the election process to interoperate. EML is based on XML.

In essence, QSL has different purpose, method, and scope in contrast with EML. Firstly, QSL is aimed at as a communication tool among the questioner, implementer, and the system to fill the gap caused by the difficult communications. QSL serves as a formalized specification to QSL compiler automatically generate e-questionnaire, e-testing, and e-voting systems. As the purpose, EML is focused on defining open, secure, standardized and interoperable interfaces between components of election systems for data exchange. EML gives a detailed structure how to transmit information helping to implement the e-voting system. Secondly, QSL is designed through the combinations of a small amount of primitive elements for more comprehensive requirement specifications. EML is designed with a terminology based on two complementary high-level process models of an election exercise. Thirdly, to distinguish from scope, QSL is used to specify e-voting systems, as well as e-questionnaire and e-testing systems. EML can only specify e-voting systems.

6 Conclusion

In this paper, we have proposed an extension of QSL for various e-voting and e-voting systems, presented two security specification snippets of Estonian System and POLYAS system by QSL. From the specifications of these 2 systems, we confirm that QSL is hopeful to specify e-voting systems.

The extended QSL can provide convenience for questioner to design e-voting systems, to communicate clearly with implementer with several desirable necessary specifications, and to reuse the data with a unified format; and also provide convenience for implementer to clearly understand what questioner needs even the implementer are not the specialist in election. Moreover, the extended QSL emphasizes the specifications of security requirements, which can be used to further improve the specifications for e-questionnaire and e-testing systems.

In the future, we will continue working on improving QSL for various e-voting and e-voting systems, and implementing a compiler of QSL to provide convenience to automatically generate e-questionnaire, e-testing, and e-voting systems.

References

1. AddPoll. http://www.addpoll.com
2. AISE Lab, Saitama University: QSL Manual (ver. 1.5) (2014). http://www.aise.ics.saitama-u.ac.jp/QSL
3. FC2 Vote (in Japanese). http://vote.fc2.com
4. Google Form. http://docs.google.com
5. Grizalis, D.A. (ed.): Secure Electronic Voting. Advances in Information Security, vol. 7. Kluwer (2002)
6. Helio. https://vote.heliosvoting.org
7. Madise, U., Martens, T.: E-voting in estonia 2005. The first Practice of country-wide binding internet voting in the world. In: Krimmer, R. (ed.) Proceedings of the 2nd International Workshop on Electronic Voting 2006, LNI GI Series, Bonn, Germany, pp. 15–26 (2006)
8. Mobo Survey. http://www.mobosurvey.com
9. OASIS Election Markup language (EML) Specification Version 7.0 (2010). http://docs.oasis-open.org/election/eml/v7.0/cs01/eml-v7.0-cs01.html
10. OQSS. http://www.oqss.com
11. Opinion Stage. http://www.opinionstage.com
12. Opoll (in Chinese). http://www.opoll.com
13. POLYAS. https://www.polyas.com
14. Poll Every Where. http://www.polleverywhere.com
15. ProProfs. http://www.proprofs.com
16. Qadah, G.Z., Taha, R.: Electronic voting systems: requirements, design, and implementation. Comput. Stand. Interf. **29**(3), 376–386 (2007)
17. QuestionPro. http://www.questionpro.com
18. Simple Voting. https://www.simplyvoting.com
19. Snappy Poll. https://www.snappypoll.com
20. Stone Poll (in Chinese). http://www.stonepoll.com
21. SurveyMonkey (in Japanese). http://www.surveymonkey.com
22. Tou Piao Wang (in Chinese). http://www.toutoupiao.com
23. Voice Poll. https://voicepolls.com
24. Volkamer, M.: Evaluation of Electronic Voting. LNBIP, vol. 30. Springer, Heidelberg (2009). doi:10.1007/978-3-642-01662-2
25. Votenet. http://www.votenet.com

26. Wang, Z., Zhou, Y., Wang, B., Goto, Y., Cheng, J.: An extension of QSL for e-testing and its application in an offline e-testing environment. In: Park, J.J., Chao, H.-C., Arabnia, H., Yen, N.Y. (eds.) Advanced Multimedia and Ubiquitous Engineering. LNEE, vol. 352, pp. 7–14. Springer, Heidelberg (2015)
27. W3C: Extensible Markup Language (XML) 1.0, 5th edn. http://www.w3.org/TR/2008/REC-xml-20081126/
28. W3C: XML Schema. https://www.w3.org/XML/Schema
29. YouPoll. http://www.youpolls.com
30. Zhou, Y., Goto, Y., Cheng, J.: QSL: A Specification Language for E-questionnaire Systems. In: Proceedings of the 5th IEEE International Conference on Software Engineering and Service Science (ICSESS 2014), Beijing, China, pp. 224–230. IEEE (2014)