QSL: A Specification Language for E-Questionnaire, E-Testing, and E-Voting Systems

Yuan Zhou, Hongbiao Gao and Jingde Cheng

Abstract E-questionnaire, e-testing, and e-voting are the essential ingredients of modern communities as the methods for a group to express a choice, a preference, or an opinion by an e-paper. Many kinds of e-questionnaire, e-testing, and e-voting systems are implemented to provide e-questionnaire, e-testing, and e-voting services on the Internet. However, there is a gap manifested in difficult communications among questioners, developers, and systems. To cover the gap, this paper proposes QSL, the first specification language with a standardized, consistent, and exhaustive list of requirements for specifying various e-questionnaire, e-testing, and e-voting systems such that the specifications can be used as the premise of automatically generating e-questionnaire, e-testing, and e-voting systems. This paper also presents QSL structure satisfying stability and extensibility, shows various QSL applications for providing convenient QSL services to questioner and developer.

Keywords QSL · Specification language · E-questionnaire · E-testing · E-voting · Web service

1 Introduction

E-questionnaire, e-testing, and e-voting are the essential ingredients of modern communities as the methods for a group to express a choice, a preference, or an opinion by an e-paper. Over a decade, there are many kinds of e-questionnaire,

Y. Zhou · H. Gao · J. Cheng (✉)
Department of Information and Computer Sciences,
Saitama University, Saitama, Japan
e-mail: cheng@aise.ics.saitama-u.ac.jp

Y. Zhou
e-mail: shuugen@aise.ics.saitama-u.ac.jp

H. Gao
e-mail: gaohongbiao@aise.ics.saitama-u.ac.jp

J.J. (Jong Hyuk) Park et al. (eds.), *Advanced Multimedia and Ubiquitous Engineering*, Lecture Notes in Electrical Engineering 393,
DOI 10.1007/978-981-10-1536-6_34

e-testing, and e-voting systems implemented to provide e-questionnaire, e-testing, and e-voting services on the Internet. According to intents and purposes by different companies, organizations, and individuals, different requirements for different types of e-questionnaire/e-testing/e-voting can thus derive different e-questionnaire/e-testing/e-voting systems.

There are four difficult problems on communications among questioners, developers, and systems. If existing e-questionnaire/e-testing/e-voting systems cannot conduct an e-questionnaire/e-testing/e-voting a questioner wants, the questioner may need a new system. On communication between questioner and developer, firstly, it is difficult for questioner to clearly describe specifications for the new system, since he/she does not know what are necessary. Secondly, it is also difficult for developer to understand the specifications for the system from a questioner because the questioner may be not able to clearly describe the specifications and developer is not a specialist in questioner's field. Thirdly, on communication between questioner and the systems, it is a big burden to learn how to use a new system if questioner does not know the usage of the system. At last, it is inconvenient that data cannot be generally reused, if a questioner wants to use downloaded data from an e-questionnaire/e-testing/e-voting system in others. It is desirable to automatically generate e-questionnaire, e-testing, and e-voting systems for satisfying what questioner wants, providing convenience, and saving labor cost and time.

Therefore, it is necessary to provide a specification language as a formalized specification. The specification language should specify necessary requirements for e-questionnaire, e-testing, and e-voting systems with a unique precise definition, because providing a specification language for both questioner and developer can make questioner to clearly describe the requirements for the system, and developer can also clearly understand what questioner needs. In addition, developer can implement the system, which provides e-questionnaire, e-testing, and e-voting services. Communication between questioner and e-questionnaire/e-testing/e-voting system, it will be easy because of only one method for questioner. Meanwhile, it is convenient that specification language provides questioner with data by a unique format that can be reused. Figure 1 shows the relationship of designer of a specification language, questioner and developer of e-questionnaire, e-testing, and e-voting systems. Besides, the ideal state is that a tool as a complier to automatically generate e-questionnaire, e-testing, and e-voting systems for questioner. With the feedbacks from questioner and developer, designer revises the language.

This paper proposes QSL, the first specification language for e-questionnaire, e-testing, and e-voting systems that serves as a formalized specification for specifying various e-questionnaire, e-testing, and e-voting systems with a standardized, consistent, and exhaustive list of requirements, such that the specifications can be used as the premise of automatically generating e-questionnaire, e-testing, and e-voting systems. The rest of the paper is organized as follows: Chap. 2 presents the primitive elements of e-questionnaire, e-testing, and e-voting systems. Chapter 3 presents QSL structure. QSL applications for providing convenient QSL services are presented in Chap. 4. Finally, some concluding remarks are given in Chap. 5.

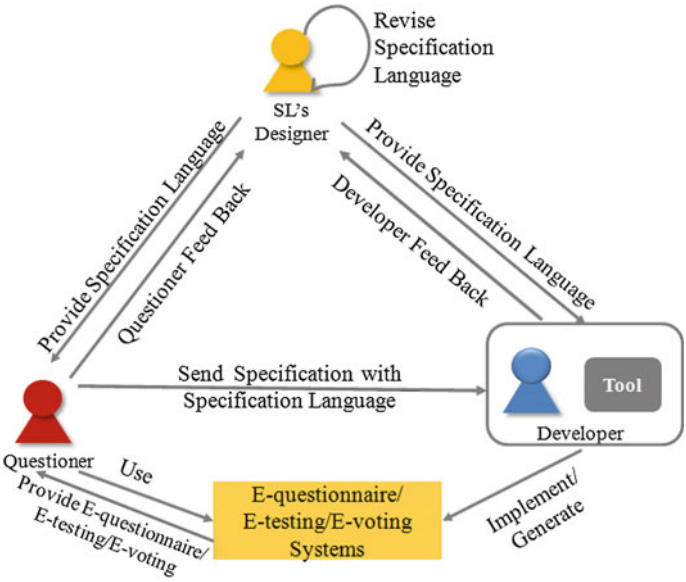

Fig. 1 Relationships among designer, questioner, and developer

2 Primitive Elements of E-Questionnaire, E-Testing, and E-Voting Systems

In order to specify various e-questionnaire, e-testing, and e-voting systems, the basic idea is to find out the primitive elements of e-questionnaire, e-testing, and e-voting systems by surveying various e-questionnaire systems and represent various e-questionnaire, e-testing, and e-voting systems by combining the elements. The primitive element is an element used to combine into various e-questionnaire, e-testing, and e-voting systems with its independent attribute, as well as the basis of design of the questionnaire specification language.

We investigated 30 e-questionnaire systems, 20 e-testing systems, and 20 e-voting systems [2, 5]. We found the similarities and differences among e-questionnaire, e-testing, and e-voting systems. We extracted elements from all the functions of the systems and classified them according to the similarities and differences. The primitive elements of e-questionnaire, e-testing, and e-voting systems consists of entity and representation. Entity is uniquely identified regardless of changing representation, which will be designed as an element in QSL. Representation is to express the corresponding entity that will be designed as the attribute or value of the element.

There are 45 primitive elements [1] classified into 6 independent groups: participant, questionnaire/examination paper/ballot (*paper* for short), function of an e-questionnaire/e-testing/e-voting (*function* for short), server, software, and phase.

3 QSL: A Specification Language for E-Questionnaire, E-Testing, and E-Voting Systems

3.1 Overview

As a specification language, we designed Questionnaire Specification Language (QSL for short), and it serves as a formalized specification for specifying various e-questionnaire, e-testing, and e-voting systems. QSL provides vocabulary and notation to describe various desirable functions of e-questionnaire, e-testing, and e-voting systems, and necessary items to make description clear and precise, and to use as format for data exchange.

QSL is based on XML [3] because XML is a versatile markup language, capable of labeling the information content of diverse data sources including structured and semi-structured documents that allow us to use the structure of XML intelligently and express the complete specification of e-questionnaire, e-testing, and e-voting systems. Furthermore, XML provides a standard way produced by W3C so that there are many tools and software to support it. Since the XML schema language is a formalization of constraints, expressed as rules or a model of structure, that apply to class of XML documents [4]. The grammar of QSL is defined by XML schema.

3.2 QSL Structure

According to the primitive elements of e-questionnaire, e-testing, and e-voting systems, we design QSL structure. Figure 2 illustrates QSL structure of relationship overview among e-questionnaire, e-testing, and e-voting.

In order to extend and upgrade QSL easily without changing the whole configuration, 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

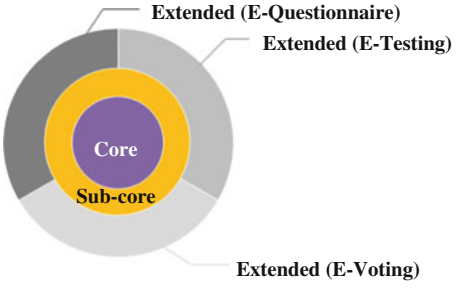

Fig. 2 Relationship overview among e-questionnaire, e-testing, and e-voting

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 the 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

Table 1 Core elements, sub-core elements, and extended elements

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-Q)	21q	Logic	Skipping, Piping, Extraction
	22q	RateControl	–
	23q	Ranking	–
Extended (E-T)	21t	Marker	Marker, Score, SampleAnswer
	22t	Score	–
	23t	SampleAnswer	–
	24t	Formula	–
Extended (E-V)	21v	Authentication	Secret, Token, Biometrics
	22v	Anonymisation	Seal, Channel
	23v	Auditing	–
	24v	CAServer	ID, Link
	25v	ATSoftware	Version, ID, Link, Solution
	26v	Candidate	ID, Name, Affiliation, Proposer

namespace defined using **QSL**. As the configuration of core elements, it gives a combination relationship of sub-core elements. In addition, some major elements for constructing sub-core elements and extended elements are shown below.

To ensure stability and extensibility, we design QSL based on its configuration, such that it provides the phase isolation to easily revise elements in middle and outermost layers without changing the general structure.

4 QSL Applications

Considering providing conveniences to both questioner and developer, it is hopeful to implement a series of applications for QSL. There are 2 kinds of applications that should be prioritized. Figure 3 illustrates QSL implementation cycle. Questioner design an e-questionnaire/e-testing/e-voting system using a specific QSL structure editor to mutually transform between requirement list and QSL specification, and to create and edit error-free QSL specifications according to QSL grammar coupled with a read-helper to hint each element's meaning. QSL structure editor output a valid QSL document. Developer can read easily this QSL document by the assistant from QSL editor, and implement an e-questionnaire/e-testing/e-voting system. A QSL compiler can read QSL document and automatically generate e-questionnaire, e-testing and e-voting systems for questioner.

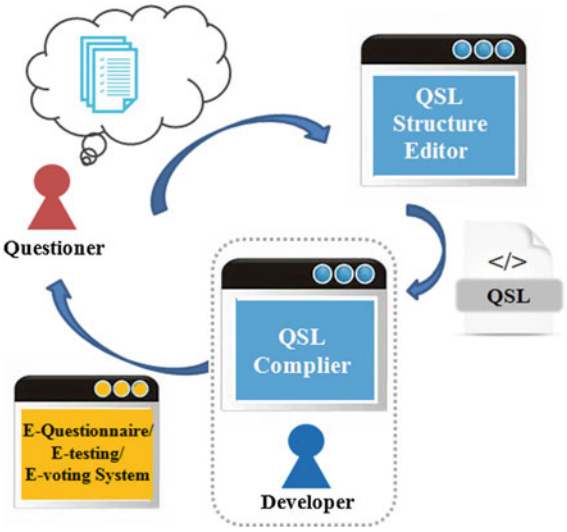

Fig. 3 QSL implementation cycle

5 Conclusion

This paper proposed QSL: a first specification language for e-questionnaire, e-testing, and e-voting systems for generating e-questionnaire, e-testing, and e-voting systems automatically. In addition, the paper presented QSL structure that satisfies stability and extensibility, and showed various QSL applications for providing convenient QSL services to questioner and developer.

In the future, we continue improving QSL through case studies. We will implement a QSL structure editor for mutual transformation between requirement list and QSL specification, and a QSL compiler system to automatically generate e-questionnaire, e-testing and e-voting systems for conveniently reusing specifications.

References

1. AISE Lab, Saitama University (2014) QSL manual (ver. 1.5). http://www.aise.ics.saitama-u.ac.jp/QSL
2. Wang Z, Zhou Y, Wang B, Goto Y, Cheng J (2015) An extension of QSL for e-testing and its application in an offline e-testing environment. In: Park JJ et al (eds) Advanced multimedia and ubiquitous engineering, Lecture notes in electrical engineering, vol 352. Springer, Berlin, pp 7–14
3. W3C: Extensible markup language (XML) 1.0 (fifth edn). http://www.w3.org/TR/2008/REC-xml-20081126/
4. W3C: XML schema. https://www.w3.org/XML/Schema
5. Zhou Y, Goto Y, Cheng J (2014) 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, NY