Doctoral Dissertation

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

電子調査システム、電子試験システム及び電子投票システムのための仕様記述言語QSL

Zhou Yuan

Graduate School of Science and Engineering,
Saitama University

Supervisor: Professor Yuichi Goto

December 2018

Abstract

Over two decades, many kinds of questionnaires, testing, and voting are performed in some completely electronic ways to do questions and answers on the Internet as Web applications, i.e. e-questionnaire systems, e-testing systems, and e-voting systems. E-questionnaire, e-testing, and e-voting systems are indispensable in society.

On the one hand, the mutual collaboration of the stakeholders is the foundation for the development and operation of e-questionnaire, e-testing, and e-voting systems. There are many stakeholders involved in a variety of e-questionnaire, e-testing, and e-voting systems around the world. To address this situation, it is necessary to promote the effective communications among those stakeholders by a communication tool to easily link each stakeholder's requirements for systems and services of e-questionnaire, e-testing, and e-voting. However, until now there is no communication tool shared among the stakeholders of systems and services of e-questionnaire, e-testing, and e-voting. All the e-questionnaire, e-testing, and e-voting systems are designed, developed, used, and maintained in various ad hoc ways. As a result, the stakeholders are difficult to communicate to implement the systems, because there is neither an exhaustive requirement list to have a grasp of the overall e-questionnaire, e-testing, and e-voting systems nor a standardized terminology for e-questionnaire, e-testing, and e-voting systems to avoid ambiguity.

A specification language is a formal language used during systems analysis, requirement analysis, and system design to describe a system. Requirements specification is the first and main step of developing software, and the specification language is the most direct way to specify requirement specifications. In addition, owing to the demands for the requirement list and the terminology, a specification language is a better communication tool to solve communication problems.

On the other hand, because e-questionnaire, e-testing, and e-voting have common processes, that is, from preparing questions, following by authenticating respondents, through submitting answers, and ending to analyzing, tallying, and declaring results, systems that provide e-questionnaire, e-testing, and e-voting services have common functions to do the processes. E-questionnaire, e-testing, and e-voting systems have lots in common.

Thus, it is necessary to provide a specification language as a communication tool to provide exhaustive requirement list and standardized terminology for systems and services of e-questionnaire, e-testing, and e-voting so that the stakeholders can communicate easily and unambiguously, and deal with and describe the requirement specifications for three kinds of systems and services. When the stakeholders are using and working with an updating system, they are looking forward to a clever and automatic way to easily deal with the changes. The ideal state is that a tool as a generator automatically converting the formalized specifications to generate the e-questionnaire, e-testing, and e-voting systems.

However, until now, there is no general-purpose specification language for specifying various e-questionnaire, e-testing, and e-voting systems in a unified format such that the stakeholders can communicate, understand and accept each other to create the requirement specifications. The existing specification languages Simple Survey System (SSS), IMS Question and Test Interoperability Specification (QTI), and OASIS Election Markup Language (EML) cannot solve the above problems, because based on their own motivations, they can neither cover e-questionnaire, e-testing, and e-voting systems nor specify exhaustive requirements of e-questionnaire, e-testing, and e-voting systems.

This thesis proposes the first specification language, named "QSL," with a standardized, consistent, and exhaustive list of requirements for specifying various e-questionnaire, e-testing, and e-voting systems. QSL can be used to provide a desirably exhaustive requirement list to the corresponding stakeholders, so that they can easily choose requirements and complete the requirement specifications.

To propose QSL, firstly, we collected, analyzed, and summarized exhaustive requirements for systems and services of e-questionnaire, e-testing, and e-voting, which are extremely similar. There are 181 requirements for e-questionnaire systems, 169 requirements for e-testing systems, and 168 requirements for e-voting systems. We summarized 121 common requirements for e-questionnaire, e-testing, and e-voting systems. The exhaustive requirements are the fundament of QSL and based on the requirements, the stakeholders can choose requirements and complete the requirement specifications easily.

Secondly, according to the investigations, we proposed QSL as a unified method to provide terminology and specifications of e-questionnaire, e-testing, and e-voting systems with a standardized, consistent, and exhaustive requirement list to solve the communication problems among the stakeholders.

Thirdly, we evaluated the proposed QSL about description power to ensure its completeness manifesting in specifying various e-questionnaire, e-testing, and e-voting systems. Current QSL can cover more than 95% system requirements and 95.4% service requirements, and provides enough notations to describe the requirements for data portability, and can cover the existing specifications compared with the related works. The reasons of the rest requirements QSL cannot cover, on the one hand, they are not necessary requirements for QSL according to the motivation, and on the other hand, they are easy to extend QSL to cover them.

Lastly, we proposed an alternative usage of QSL for data portability and implemented two real applications for QSL. QSL is used as a data format for data portability for a general-purpose e-questionnaire server to help conduct an e-questionnaire on different e-questionnaire systems, and for an offline e-testing environment to provide users with offline e-testing service to execute various offline e-testing. The applications can prove QSL is practical and available to be used to provide data format for data portability.

Acknowledgments

First and foremost, I would like to show my deepest gratitude to my supervisor, Professor Yuichi Goto, a respectable, responsible and resourceful scholar, who has provided me with valuable guidance in every stage of the writing of this thesis and help me pursue my doctoral degree.

I shall extend my thanks to Professor Jingde Cheng for all his kindness and help. Without his enlightening instruction, impressive kindness and patience, I could not have completed my thesis. His keen and vigorous academic observation enlightens me not only in this thesis but also in my future study and research.

I would also like to thank all my teachers who have helped me to develop the fundamental and essential academic competence. My sincere appreciation also goes to the members from Advanced Information Systems Engineering Laboratory, who participated in this research with great cooperation. I'd like to thank all my family and my friends, for their encouragement and support.

Last but not least, I am very grateful to my dissertation committee: Professor Norihiko Yoshida, Professor Noriaki Yoshiura, and Associate Professor Jun Ohkubo for their support, invaluable feedback, and helpful advice to this research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Questionnaire and voting are the essential activities of the modern communities as the general and indispensable methods for a group of people to express a choice, a preference, or an opinion [9]. Testing also is a necessary activity as an integral way to be widely used to assess people's achievement, ability, and characteristics [27]. Over two decades, many kinds of questionnaires, testing, and voting are performed in some completely electronic ways to do questions and answers on the Internet as Web applications, i.e., e-questionnaire systems, e-testing systems, and e-voting systems (e-questionnaire, e-testing, and e-voting systems for short). E-questionnaire, e-testing, and e-voting systems play an important role in modern society, have a unique value and are indispensable in society. If these systems are unreliable, lower security, strange in use, it will have a serious impact on society. There is still an important research topic of how to design, develop, maintain, and operate reliable, highly secure, and user-friendly e-questionnaire, e-testing, and e-voting systems.

On the one hand, the mutual collaboration of the stakeholders is the foundation for development and operation of the information systems [53]. There are five kinds of stakeholders of e-questionnaire, e-testing, and e-voting systems, which are sponsor, evaluator, executor, respondent, and supporter. Firstly, a sponsor launches activities and usually focuses on the strategic goals, return on investment, as well as the costs and time involved in building and operating the systems. Secondly, an evaluator monitors whether the system meets standards, laws, and regulations. Thirdly, an executor performs tasks to help carry out the activities after system deployment that perhaps contains questioner, monitor, analyst, etc. Fourthly, a respondent is to answer a questionnaire, a test, or a vote. Fifthly, a supporter provides the tech-support services that perhaps contains including communicator, developer, maintainer, manufacturing engineer, supplier, customer service, etc. From the respective of software engineering, the most important goal of the implementation of the e-questionnaire, e-testing, and e-voting systems is to satisfy the requirements of the stakeholders. There are many stakeholders involved in a variety of e-questionnaire, e-testing, and e-voting systems around the world. To address this situation, it is necessary to promote the effective communications

among those stakeholders by a communication tool to easily link each stakeholder's requirements for systems and services of e-questionnaire, e-testing, and e-voting.

However, until now there is no communication tool shared among the stakeholders of systems and services of e-questionnaire, e-testing, and e-voting. All the e-questionnaire, e-testing, and e-voting systems are designed, developed, used, and maintained in various ad hoc ways. Firstly, it is not so easy for the stakeholders to define exhaustive requirements of the systems and the services of e-questionnaire, e-testing, and e-voting. For instance, the sponsor, the executor, and the supporter only focus on their own requirements but have a shallow understanding of the overall requirements. It lacks an exhaustive requirement list to guide and assist the stakeholders to know overall requirements for systems and services of e-questionnaire, e-testing, and e-voting. Secondly, it is difficult to avoid ambiguity for the stakeholders because they do not have the standardized terminology on the systems and the services of e-questionnaire, e-testing, and e-voting. For example, the sponsor uses different terminologies for a common requirement of three kinds of services. It may lead the supporter to provide different solutions for the system, after all, the supporter is not the professional expert in the business fields, and the sponsor is not a technical specialist. Moreover, terminologies are not unified although the systems and the services have common processes. In another example, the sponsor and the executor use the same terminology for different requirements of services of e-questionnaire, e-testing, and e-voting. It may lead the supporter to provide the same solution for the system. It lacks a standardized terminology to unify the common requirements and to normalize the different requirements for services of e-questionnaire, e-testing, and e-voting and the corresponding systems.

A specification language [13] is a formal language in computer science used during systems analysis, requirement analysis, and system design to describe a system. From the viewpoint of software engineering, requirements specification is the first and main step of developing a software, and so the specification language is the most direct way to specify requirement specifications, after all, the requirement specification is the result by communicating among the stakeholders. In addition, owing to the demands for exhaustive requirement list and the terminology, a specification language is a better communication tool to solve communication problems.

On the other hand, because e-questionnaire, e-testing, and e-voting have common processes, that is, from preparing questions, following by authenticating respondents, through submitting answers, and ending to analyzing, tallying, and declaring results, systems that provide e-questionnaire, e-testing, and e-voting services have common functions to do the processes. E-questionnaire, e-testing, and e-voting systems have lots in common. In addition, in fact, some representative systems [1, 10, 17, 35, 38, 49, 52, 58, 62] exist to provide three-in-one service for people all over the world.

Thus, it is necessary to provide a specification language as a communication tool to provide exhaustive requirement list and standardized terminology for systems and services of e-questionnaire, e-testing, and e-voting so that the stakeholders can communicate easily and unambiguously, and deal with and describe the requirement specifications for three kinds of systems and services. Fig. 1.1 shows

the relationship between the specification language and the stakeholders. The specification language allows the sponsor, the executor, the evaluator, and even the respondent to have an overall consciousness owing to the exhaustive requirement list, and then to use it to easily communicate with each other as well specify the requirement specifications for three kinds of services. Because of the standardized terminology to avoid ambiguity, they can communicate clearly with the supporter. Referring to the service requirement specifications, the supporter can specify the system requirement specification based on the specification language and even implement the system. Meanwhile, it is convenient that the specification language provides the sponsor, the executor, and the respondent with data by a unique format that can be reused. Besides, when the stakeholders are using and working with the updating system, which needs to add items, change functions, and add constraints, they are looking forward to a clever and automatic way to easily deal with the changes. The ideal state is that a tool as a generator automatically converting the formalized specifications to generate the e-questionnaire, e-testing, and e-voting systems.

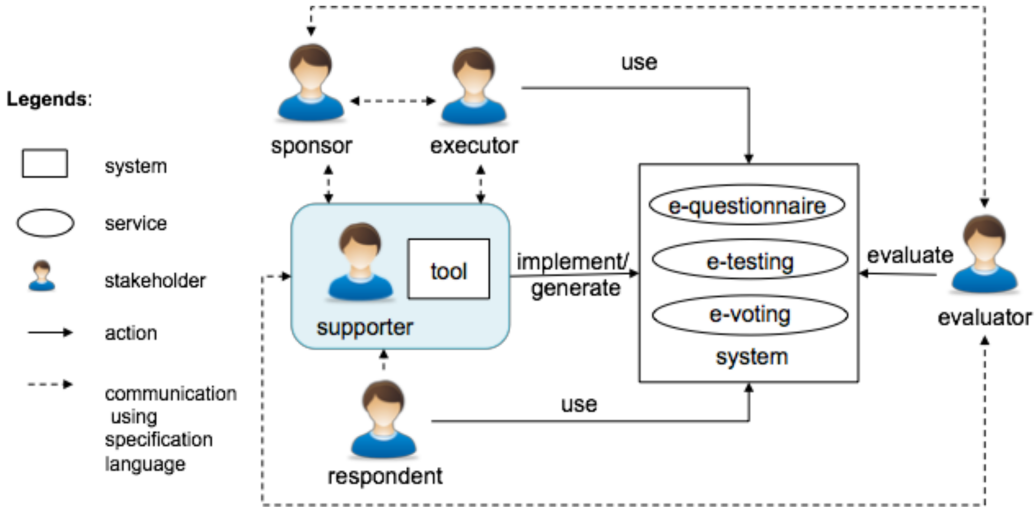

Figure 1.1: Relationships of specification language and stakeholders.

However, the existing specification languages Simple Survey System (SSS) [8], IMS Question and Test Interoperability Specification (QTI) [29], and OASIS Election Markup Language (EML) [40] cannot solve the above problems, because based on their own motivations, they can neither cover e-questionnaire, e-testing, and e-voting systems nor specify exhaustive requirements of e-questionnaire, e-testing, and e-voting systems. And that means until now, there is no general-purpose specification language can provide precise and essential description ways for specifying various e-questionnaire, e-testing, and e-voting systems in a unified format such that the stakeholders can communicate, understand, and accept each other to create the specifications of the essentially similar systems for e-questionnaire, e-testing, and e-voting.

1.2 Purpose and Objectives

The purpose is to design the first specification language for e-questionnaire, e-testing, and e-voting systems that serves as a communication tool for specifying various e-questionnaire, e-testing, and e-voting systems with a standardized, consistent, and exhaustive list of requirements.

To accomplish the purpose, our objectives are

- to analyze exhaustive requirements of e-questionnaire, e-testing, and e-voting systems;
- to define QSL addressing that it can specify all the requirements of various e-questionnaire, e-testing, and e-voting systems;
- to evaluate about the descriptive power of QSL manifesting in specifying various e-questionnaire, e-testing, and e-voting systems;
- to do case studies for the specification language.

1.3 Structure of This Thesis

In this thesis, we propose the first specification language, named "QSL," [4, 72, 76, 77, 78, 79] 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 precondition of automatically generating e-questionnaire, e-testing, and e-voting systems.

This thesis is organized as follows. Chapter 1 presents the background, motivation, and purpose of this research. Chapter 2 presents an analysis of the exhaustive requirements of e-questionnaire, e-testing, and e-voting systems. Chapter 3 provides the QSL definitions addressing that it can specify all the requirements of various e-questionnaire, e-testing, and e-voting systems in a structured way for satisfying its stability and extensibility. Chapter 4 presents an evaluation about the descriptive power of QSL manifesting in specifying various e-questionnaire, e-testing, and e-voting systems. Chapter 5 presents real applications of QSL in case of e-questionnaire, e-testing, and e-voting systems we implemented. Some conclusions are given in Chapter 6.

Chapter 2

Exhaustive Requirements of E-Questionnaire, E-Testing, and E-Voting Systems

2.1 Collecting Requirements of E-Questionnaire, E-Testing, and E-Voting Systems

QSL is proposed to provide exhaustive requirements list of e-questionnaire, e-testing, and e-voting systems to the stakeholders so that they can design, develop, and evaluate their demanded systems easily. In order to get exhaustive requirements of e-questionnaire, e-testing, and e-voting systems, at first, we investigated 29 e-questionnaire systems [1, 2, 5, 10, 11, 12, 14, 15, 16, 17, 22, 23, 31, 35, 36, 37, 38, 44, 49, 51, 52, 55, 56, 58, 59, 62, 63, 64, 65], 22 e-testing systems [1, 10, 12, 17, 19, 20, 30, 33, 35, 38, 41, 42, 44, 46, 49, 50, 52, 58, 59, 62, 66, 74], and 23 e-voting systems [1, 10, 12, 17, 21, 25, 28, 35, 38, 44, 45, 47, 49, 52, 54, 57, 58, 61, 62, 67, 68, 71, 75], which are representative systems seizing a large number of high quality customers all over the world for serving a relatively long time. And we enumerated and summarized the requirements from these systems.

Afterwards, we found e-questionnaire, e-testing, and e-voting systems have some common and specific requirements. Fig. 2.1 illustrates a Venne diagram to present the relationship of e-questionnaire, e-testing, and e-voting systems. The three primary color circles stand for the sets of requirements of e-questionnaire, e-testing, and e-voting systems, respectively. The union of four numbered regions in a circle presents a complete system. For instance, a set of requirements for a complete e-questionnaire system is a union of requirements in region 1, 2, 3, and 7.

Region 1 lists the intersection of the sets of requirements of e-questionnaire, e-testing, and e-voting systems, i.e., a set of common requirements of e-questionnaire, e-testing, and e-voting systems. The common requirements are core contents to construct a general-purpose e-questionnaire, e-testing, and e-voting system. The common requirements of e-questionnaire, e-testing, and e-voting systems are summarized for two aspects: system, and service. Firstly, there are 55 common re-

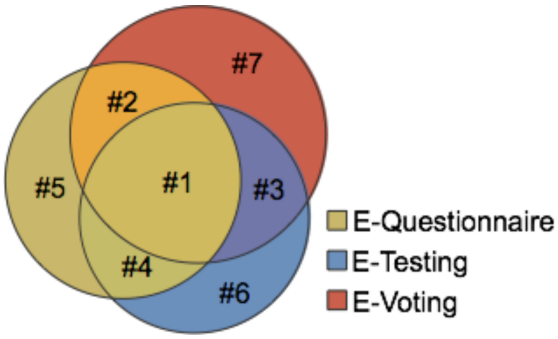

Figure 2.1: Relationship of requirements of e-questionnaire, e-testing, and e-voting systems.

quirements for the system of e-questionnaire, e-testing, and e-voting. All these three kinds of systems have common phases, which are *setting-up* to prepare e-questionnaire, e-testing, and e-voting; *distributing* to distribute the paper to respondents; *registering* to allow anyone through some authentication check to be an eligible respondent;*submitting* to answer the questions in the paper and send to submitting server, usually called voting phase in e-voting; *collecting* to collect the responses of the respondents; *analyzing* to analyze the collected responses; and *counting* to calculate the analyzed responses and get results, usually called tallying phase in e-voting [70]. Based on the phases, they have common security requirements such as *authority* to ensure the access control of the participants, and common function requirements such as launching the paper or stopping launching during setting up phase, as well as common environment requirements such as server to store the collected results after submitting phase and server to provide registration services for respondents in registering phase. Secondly, there are 66 common requirements for the service of e-questionnaire, e-testing, and e-voting systems. These three kinds of services should provide common questionnaire structure, which is a question sheet on a form to express a choice preference, usually called ballot in e-voting. It consists of section, question, answer, and some pictures and videos to represent it well recorded as *media*, etc. There are some common requirements for question types such as multiple choices, validation common requirements to limit the responses, to control the required response, etc., and common setting requirements such as *language* to support the representation of multi-language. The services are inseparable from the common roles of the participants, which are *sponsor* who organizes and supports an event; *questioner* who designs a paper and settings, usually called examiner in e-testing; *analyst* who processes the collected responses; *monitor* who monitors whether illegal or dishonest behavior occurs or not; *respondent* who answers the questions in the paper, usually called examinee in e-testing and voter in e-voting [27]; and *analyst* who analyze the collected responses. In addition, these services have common requirements for recording information such as responses from respondents, and analyzed report.

Region 2, region 3, and region 4 list the intersections of two systems. In region 2, both e-questionnaire and e-voting systems have same question types such as

contact information to record the name, address, fax, phone number, and so on. It is an advanced question type of open-ended text, which derives from single row text to present in multiple lines. Region 3 lists the similar requirements of e-testing system and e-voting system. Both e-testing and e-voting systems have similar security requirements considering on ensure fairness of respondents. As to region 4, both e-questionnaire and e-testing have some common logic requirements.

Region 5, region 6, and region 7 list the specific requirements of e-questionnaire, e-testing, and e-voting systems. In region 5, e-questionnaire differs with others in its complex question types and complex logic types. E-questionnaire has four basic question types, which are multiple choice, open-ended text, matrix, and ranking, as well as two possible combination question types. Each question type corresponds with limitation and arrangements derive widely variety of question types that e-testing and e-voting are unnecessary. In addition, e-questionnaire ensures that only relevant questions are displayed to the appropriate respondents according to its complex question types. E-questionnaire has four basic logic types, which are skipping, piping, extraction, and randomization, as well as three kinds of possible combinations of basic logic types. In region 6, *marking* is a phase to mark the responses and give the scores referring to the sample answers. The score of each answer and a total score of answers are demanded. Besides, questions in e-testing involve wider and more professional field, such as mathematical and chemical formula, etc. As to region 7, the differences are around security, because e-voting is mainly used in governmental elections for the universal, equal, free, and secret suffrage. E-voting has *auditing* phase to check whether the result is correct. E-voting also needs certification server to provide services to validate the respondents to prevent any possibility of affecting results. In addition, e-voting needs an auditing software to communicate with submitting system. In addition, e-voting needs a list of candidates, and provides a candidate nomination and candidate registration.

In addition, we classified these requirements for services into 8 groups, which are phase, paper, question type, logic, validation, setting, data, and participant. And the requirements for systems are classified into 3 groups, which are environment, function, and security. The groups are the core models to construct a general e-questionnaire, e-testing, e-voting system. Table 2.1 presents brief and pithy requirements mapping to the corresponding regions of e-questionnaire, e-testing, and e-voting systems, respectively, as well as mapping to the corresponding groups of requirements in each region. Considering limited spaces, the detailed requirement lists for systems and services of e-questionnaire, e-testing, and e-voting can be referred to Appendix B.

Through the investigation, we summarized 181 requirements for e-questionnaire systems, 169 requirements for e-testing systems, and 168 requirements for e-voting systems. There are 121 common requirements for e-questionnaire, e-testing, and e-voting systems. These requirements assist to define descriptions and boundaries of each system and service. According to the features of requirements, we classified these requirements for services into eight groups, which are phase, paper, question type, logic, validation, setting, data, and participant. And the requirements for systems are classified into three groups, which are environment, function, and

security.

2.2 Terminology

In order to define the terminology for QSL, we extracted the keywords from the summarized requirements, analyzed the keywords, picked up them who have an independent feature in spite of a variety of representations, and defined them as entities. A set of entities is, most tangibly, a set of real objects that cannot be disintegrated, and can be combined in varying amounts describing a gamut of requirements for e-questionnaire, e-testing, and e-voting systems. This is the essential method used to be intended to elicit the descriptions of diverse requirements for e-questionnaire, e-testing, and e-voting systems. The entities are suitable as terminology for QSL.

There are 54 entities arranged in Fig.2.2. The column stands for a group of entities. The groups listed respectively correspond to *phase*, *participant*, *function*, *environment*, *security*, *paper*, *setting*, and *data*.

Phase	Participant	Function	Env.	Security	Paper	Setting	Data
setting up	sponsor	export	server	authority	paper	language	sample
distributing	questioner	import	gateway	authentication	description	time	score
registering	respondent	launch	interface	anonymity	section	number	response
submitting	analyst	stop	device	seal	question	limitation	result
collecting	monitor	generate	software	channel	answer		field
analyzing	marker	ping			media		
counting	auditor	integrate			logic		
marking	candidate	remind			alignment		
auditing	proposer				formula		

Figure 2.2: A form of entities.

The first group lists the entities to describe each phase. Marking is a phase in e-testing, and auditing is a phase in e-voting. Phases relate to service's settings and system's functions. In the second group, it lists the entities to describe each role of participants. Both candidate and proposer are the roles of the participant in e-voting. A candidate relates with a number of proposers in an election. The third group shows the entities to describe functions. Some functions are demanded during the whole phases. For example, the system provide functions for a questioner to export the paper file in the whole phases, to import the paper file in setting up phase, to launch and stop the survey after setting up phase, and even generate registration tickets to respondents during registering phase, ping IP for monitors before submitting phase, integrate responses and send to analysts during collecting phase, and remind before counting phase for respondent whom did not

take part in the survey approaching deadline. The fourth group lists the entities to describe software and hardware for the environment. In the fifth group, it lists the entities to describe security. The sixth group lists the entities to describe the paper sheet. Considering the question type is a feature for question entity, we defined question type as the representation of the question entity. As same as logic type, it is the representation of the logic entity. The seventh group lists the entities to describe settings. As to validation to describe the limitations for the settings, so we defined it as limitation entity. The last group lists the entities to describe data. The field is a data entry in a record for the system database. More to the point, we define these 8 groups of the entities, but there is no conflict with 11 groups of the summarized requirements. The 8 groups are also suitable as terminology for QSL.

2.3 Summary

This section elaborates on investigating and analyzing the exhaustive requirements of e-questionnaire, e-testing, and e-voting systems, summarizing the common requirements and specific requirements of e-questionnaire, e-testing, and e-voting systems, and classifying the requirements into 11 groups. The common requirements and the groups are the core to construct a general e-questionnaire, e-testing, e-voting system. In addition, in this section, we present the extraction of keywords as entities for combining an exhaustive requirements list for various e-questionnaire, e-testing, and e-voting systems. And we show the classifications of the entities in 8 groups. These entities and the groups of entities are suitable as the terminology for QSL. In general, owing to the relationships of requirements of e-questionnaire, e-testing, and e-voting systems, and independence of the entities, it builds the foundation for the design of a specification language of various e-questionnaire, e-testing, and e-voting systems.

Table 2.1: A list of requirements of e-questionnaire, e-testing, and e-voting systems.

Region	**Group**	**Requirements**
#1	phase	setting up, distributing, registering, submitting, collecting, analyzing, counting
	paper	section, question, answer, media, etc.
	question type	multiple selections, open-ended text, drop-down list, image chooser, rank order, etc.
	validation	limitation, response required, etc.
	setting	time, language, numbering, etc.
	environment	device, server, software, database, etc.
	participant	sponsor, questioner, respondent, etc.
	data	response, report, participant information
	function	launch, stop launching, distribute, etc.
	security	participant authority, authentication, blind paper for anonymity, etc.
#2	question type	contact information, data reference
#3	environment	security software, mix net, etc.
	function	confirm response
	security	token authentication, HSM for anonymity, etc.
#4	question type	matrix multiple choices, image hot spot
	logic	skipping, randomization, etc.
	setting	question analysis, distribution method, etc.
	function	import paper, export response, etc.
#5	question type	rating, multiple dimensions matrix, etc.
	validation	quota, weighting, etc.
	logic	extraction, complex piping etc.
	setting	trend analysis, balancing, etc.
	environment	analyst-oriented device
#6	phase	marking
	question type	formula, connect point, etc.
	validation	scoring standard
	environment	marking server, marking software, etc.
	participant	marker
	data	score, marker information
	function	mark response
	security	blinded mark
#7	phase	auditing
	environment	option nomination server, database, etc.
	participant	auditor, candidate, proposer
	data	candidate information, no-vote reason, etc.
	function	store first vote, launch proposal, etc.
	security	biometric authentication, holomorphic encrypts for anonymity, etc.

Chapter 3

QSL: A Specification Language for E-questionnaire, E-testing, and E-voting Systems

3.1 Overview

Questionnaire Specification Language (QSL) serves as a formalized specification for specifying various e-questionnaire, e-testing, and e-voting systems. Current QSL is version 3.1 [4]. QSL is based on Extensible Markup Language (XML) [73]. The grammar of QSL is defined by XML Schema [69]. According to summarized exhaustive requirements, using QSL can specify complete specifications for e-questionnaire, e-testing, and e-voting systems, so that can solve the communication problems among the stakeholders.

QSL can be used in three ways. Firstly, QSL can be used to specify e-questionnaire, e-testing, and e-voting systems. In other words, QSL can be used to specify the requirements the system provides. For example, QSL provides tags for the server to register the respondents' information, submit and analyze responses to get results. Secondly, QSL can be used to specify e-questionnaire, e-testing, and e-voting on the system, i.e., QSL can be used to describe restrictions in each phase of the e-questionnaire, e-testing, and e-voting. For example, QSL can be used to describe the distribution method such as e-mail, web-link, or other offline methods. Thirdly, QSL can be used as format for portable data of e-questionnaire [32], e-testing, and e-voting. Formats of data of e-questionnaire, e-testing, and e-voting systems are the parts of specifications of e-questionnaire, e-testing, and e-voting systems. Portable data [6] is formatted according to a published syntax and where the metadata is explicit, either included with the data or by reference to an open technical dictionary. For instance, QSL is used to describe various questions, responses, and results.

3.2 QSL Components

QSL consists of three kinds of components, which are QSL template, QSL schema, and ontology. 3.2 presents the relations and usages of QSL components.

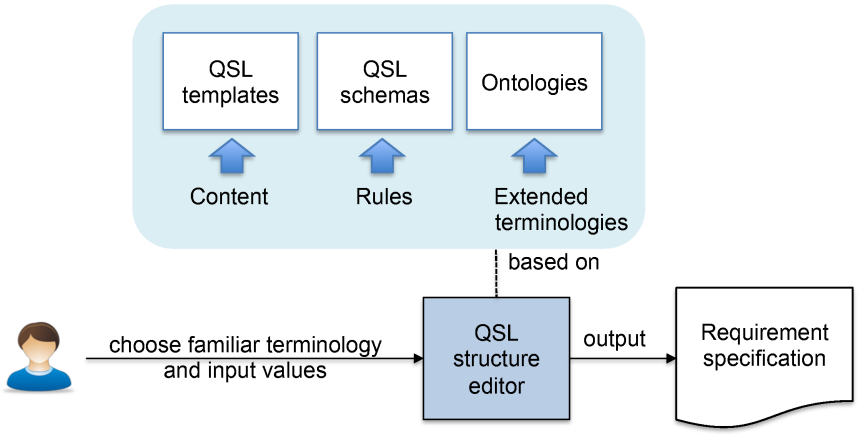

Figure 3.1: Relations and usages of QSL components.

Before the stakeholders finish the requirement specification, they should be provided with a complete list of requirements, that is, a requirement specification without values. **QSL template** is a QSL-format requirement list. It displays the concrete contents of the list of requirements. There are 65 QSL templates, and 9 of them are commonly used. Actually, a complete requirement template is rarely used, because different roles of stakeholders focus on different kinds of requirement lists. In general, from the viewpoint of the supporters, they are concerned with the system requirements. Executors pay more attention to service requirements. In particular, questioners commonly need a list for help construct a paper sheet and corresponding setting. Analysts are desirable to have a list for all the responses. Sponsors and monitors follow a list for a participant sheet. As to the developers, the suppliers, and evaluators focus on function requirements, environmental requirements, and security requirements, respectively. QSL templates are aimed to orient to different stakeholders for providing their required requirement lists.

The template conforms to the rule, which is defined and called as **QSL schema**. The schema is a skeleton or a part of the template that constrains where are tags in the document, which attributes they have, which kinds of data are filled in, and so on. There are 66 QSL schemas. QSL schema is the core component of QSL. Any QSL schema can be converted to corresponding QSL template by any XML parsing and will be detailed in the next section.

Both QSL templates and QSL schemas use a unified terminology in order to minimize any trouble about remember too many terminologies. We explicitly summed up terminologies for e-questionnaire, e-testing, and e-voting systems into one, when we design QSL. **Ontology** is an extended terminology to clearly define the relations of e-questionnaire, e-testing, and e-voting systems in order to improve QSL's usability so that the stakeholders can communicate with each other

and write specifications by familiar terminologies. There are 3 ontologies for e-questionnaire, e-testing, and e-voting systems, respectively.

The stakeholder chooses a QSL template, chooses an ontology to use a familiar terminology that is instead of original one, and put values according to the limitation of corresponding QSL schema. The output is a QSL-format requirement specification. A QSL structure editor is a supporting tool to help easily write the requirement specification.

3.3 QSL Structure

QSL structure is a regular pattern, which analyzes from requirement materials through requirement analysis for e-questionnaire, e-testing, and e-voting systems. QSL structure is the pattern indicating the corresponding relations between requirements and tags, the order of requirements, the hyponymy relations of requirements, the conditions that occur of requirements. It is the framework of the language. We define the QSL structure as a series of QSL schemas, which is the core component of QSL and the rule and fundament of the requirement specifications for e-questionnaire, e-testing, and e-voting systems.

A tag is used to describe a piece of data for requirements of e-questionnaire, e-testing, and e-voting systems. QSL provides 93 tags as terminology because QSL is an XML-based specification language. We defined two kinds of tags for QSL. Firstly, 54 tags are defined according to entities. Using these tags, we can describe requirements of systems and services of e-questionnaire, e-testing, and e-voting formally. Since XML documents have a hierarchy structure. From the viewpoint of this construction, those tags have some hierarchical relationships. Secondly, 38 tags are defined according to the construction. In addition, QSL provides 52 complex types of the hierarchy of the tags and 34 simple type for the constraint of the tags.

In QSL structure, each schema provides an intuitive means of requirement navigations for different systems and services for e-questionnaire, e-testing, and e-voting by organizing the corresponding tags in a hierarchical structure. To assist the stakeholders choose the corresponding schemas, we defined each schema and marked with number. Firstly, 100-series schemas are used to declare QSL document and define the structures of system template and service template. The stakeholder can easily get a template with QSL declaration. Secondly, 200-series schemas are used to define the subdivisions of the structures of templates according to the groups of entities we summarized previously, these schemas help the stakeholders clearly understand what kinds of requirements should be specified in a template. Thirdly, 300-series schemas are used to define the structures of skeletons of common requirements. Fourthly, 400-series, 500-series, and 600-series schemas are used to define the structures of skeletons of specific requirements of systems and services for e-questionnaire, e-testing, and e-voting, respectively. Using the marked numbers, the stakeholders can easily distinguish these three kinds of systems and services. Overall, we defined 65 schemas in QSL structure. There are 44 schemas (300-series schemas) for specifying common requirements,

10 schemas (400-series, 500-series, and 600-series schemas) for specifying specific requirements, 10 schemas (110, 120, and 200-series schemas) for constructing the hierarchy structure of QSL documents, and 1 schema (100 schema) for QSL document declarations.

Table 3.1 and Table 3.2 show a list of core schemas. Core schemas marked before reaching 400 are used to describe the structures of skeletons of requirements for constructing a basic and nondistinctive e-questionnaire, e-testing, and e-voting system.

Table 3.1: Core schemas configuration (1).

Code	**Schema**	**Purpose**
100	QSL	to declare QSL document
110	System	to specify system template
120	Service	to specify service template
210	Phase	to specify sequence of all the phases
220	Security	to specify security requirements
230	Paper	to specify paper sheet
240	Setting	to specify settings for paper sheet
250	Environment	to specify software, hardware, network, etc.
260	Participant	to specify all the participant roles
270	Data	to specify recorded data and database fields
280	Function	to specify function requirements
310	SettingUp	to specify setting up phase
311	Distributing	to specify distributing phase
312	Registering	to specify registering phase
313	Submitting	to specify submitting phase
314	Collecting	to specify collecting phase
315	Analyzing	to specify analyzing phase
316	Counting	to specify counting phase
320	Sponsor	to specify sponsor information
321	Questioner	to specify questioner information
322	Respondent	to specify respondent information
323	Analyst	to specify analyst information
324	Monitor	to specify monitor information
330	Export	to specify export function
331	Import	to specify import function
332	Launch	to specify launch function
333	Stop	to specify stop launching function
334	Generate	to specify generate function
335	Ping	to specify ping Ip function
336	Integrate	to specify integrate function
337	Remind	to specify remind function

110 schema is oriented to the supporters and 120 schema is oriented to the executors and the sponsors. The executors and the sponsors choose 120 schema as

Table 3.2: Core schemas configuration (2).

Code	Schema	Purpose
340	Server	to specify server information
341	Field	to specify database fields
342	Gateway	to specify network information
343	Interface	to specify interface
344	Device	to specify device information
345	Software	to specify software information
350	Authority	to specify authority methods for participants
351	Authentication	to specify authenticate methods
352	Anonymity	to specify anonymous methods
353	Seal	to specify encryption information
354	Channel	to specify communication channel
360	Section	to specify section for a group of questions
361	Question	to specify question a group of answers
362	Answer	to specify answer contents
363	Description	to specify title, summary, and tips
364	Media	to specify media information for paper
365	Alignment	to specify alignment for paper
366	Limitation	to specify paper limitation
370	Language	to specify multi-languages
371	Time	to specify time settings
372	Number	to specify ordering settings
373	Quota	to specify limitation for respondent numbers
374	Response	to specify response data
375	Result	to specify result data

the structure of service template and they can complete the service requirement specification based on this schema. The supporters choose 110 schema as the structure of system template, and they refer to the service requirement specification to complete the system requirement specification. The 110 schema and 120 schema must declare QSL document that is based on the 100 schema. The remaining schemas are used to construct the 110 schema and 120 schema. In addition, both 110 schema and 120 schema can be used to provide descriptions to specify data. 110 schema provides the supporters with descriptions for database schema that is defined as 341 schema in QSL. According to 341 schema, the supporters can describe and design database fields. In addition, 120 schema is used to describe participant information, responses, and results. It is desirable to reuse these data.

Table 3.3 shows a list of specific schemas. Specific schemas are used to describe the structures of skeletons of specific requirements of systems and services for e-questionnaire, e-testing, and e-voting. The marked numbers for schemas are used to ensure recognition of which is for e-questionnaire, e-testing, or e-voting, even if the stakeholders were not familiar with the associated business.

When we version up QSL, it is easy to revise the schemas without changing the

Table 3.3: Specific schemas configuration.

Code	**Schema**	**Purpose**
410	Logic	to specify logic methods, conditions, and routes
510	Marking	to specify marking phase
520	Score	to specify score rules
530	Sample	to specify sample answer for each question
540	Formula	to specify formulas of math, chemical, etc.
550	Marker	to specify marker information
610	Auditing	to specify auditing phase
620	Candidate	to specify candidate information
630	Proposer	to specify proposer information
640	Auditor	to specify auditor information

general structure. Because according to the entities and the groups of entities, the schemas defined separately and independently are easy to be extended, revised, and added. QSL structure is a stable structure and has a well-designed extensibility mechanism.

Fig. 3.2 illustrates the inclusion relations of schemas in QSL hierarchy structure. The black rectangle stands for the schema marked with a number. The schemas in the gray rectangle are used to specify the data of systems and services of e-questionnaire, e-testing, and e-voting for data portability. 100 schema is used to declare a QSL document that must set a fixed version number. In these schemas, we defined some attributes to coordinate clear descriptions. On account of the repeatability of attribute values, we defined the schema for some simpleType elements [69] to specify the constraints for them. This schema marked as 710 are used and included with 300-series, 400-series, 500-series, and 600-series schemas.Most visibly, both 240 schema and 280 schema include 210 schema. Previously discussing, phases are essential for systems and services of e-questionnaire, e-testing, and e-voting. The stakeholders can easily identify three kinds of systems and services from the whole phases, and easily recognize functions or settings during a certain phase. Each phase schema should include function schemas and setting schemas. Considering validity and repeatability of schemas, 310 schema is defined to include these schemas, and other phase schemas add 310 schema. Thus, as the proposed QSL template is combined by identifiable, distinct schemas that the stakeholders can specify independently of others, QSL structure has well-formed stability and extensibility.

Table3.4 shows the QSL versions. Each version not only contains the schema files, but also the manual for using QSL. We are still improving QSL. From this table, it also proves that QSL has extensibility.

In summary, QSL schemas can be used to help to specify exhaustive requirements owing to stability and extensibility of QSL's well-formed structure that contributes to the stakeholders to easily communicate with each other by using QSL to describe the requirements of systems and services of e-questionnaire, e-testing, and e-voting. In addition, QSL provides data template and as a unified data format

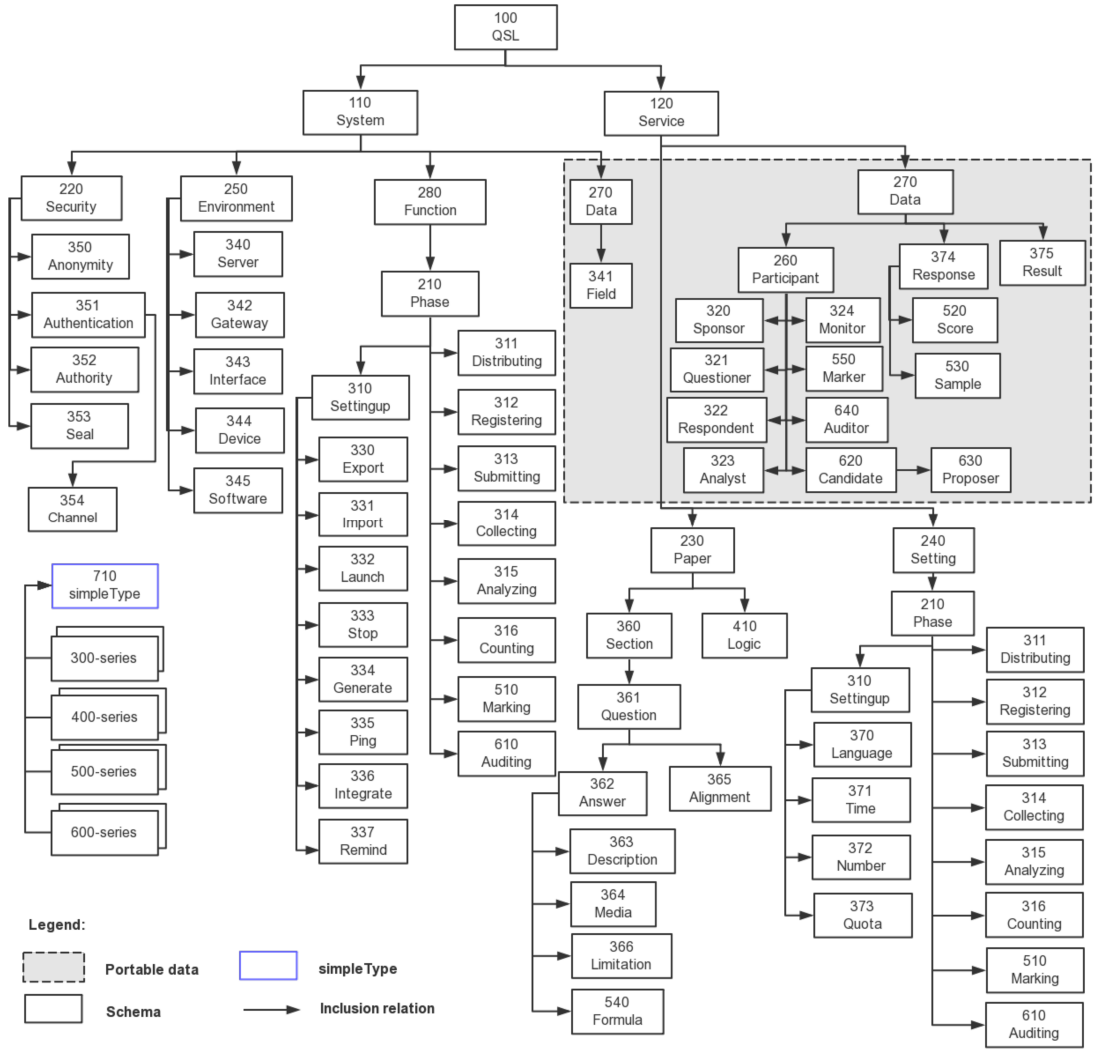

Figure 3.2: Inclusion relation among QSL schemas.

for portable data among different e-questionnaire, e-testing, and e-voting systems that contributes to improving data portability for providing conveniences for the stakeholders to reuse data.

3.4 An Example of QSL

We give an example about snippet specification using QSL system template. It specifies a general-purpose offline e-testing system we developed [72]. The specification lists the functions in each phase. Our system can import paper data and setting data described by QSL. This file is distributed using a USB flash memory. The respondents log in and are verified to ensure whether he is eligible or not. And the respondent answers the questions and submits his response. During the submitting phase, the monitor can monitor the whole phase by ping the respondents' IP address and monitor their test states. After submitting, the monitor can

Table 3.4: QSL versions.

Date	Version	Contents and changes
2014.02	1.0-1.2	First QSL proposal and prototype of QSL for e-questionnaire systems
2014.09	1.3	QSL foundation in a frame
2014.11	1.4	Add some complex question types and logic types
2014.12	1.5	Add formula, interface, functions
2015.01	1.6	Change QSL structure into 3 parts: system, paper, and security
2015.05	1.7	Add some details of phases, security, etc.
2015.11	1.8	Extend for e-testing systems
2016.08	2.0	Change new QSL structure for common and specific
2016.12	2.1	Extend some details for e-voting systems
2017.07	3.0	Change new QSL structure to desperate the simpleType and ComplexType for reuse
2018.08	3.1	Extract entities as the 300-series schemas, and cover more than 30 descriptions of requirements

use the USB to store the collected responses by a zip method. The marker can mark the responses. In addition, the corresponding components are listed such as servers, devices, database, network, and interface.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <QSL version="3.1" xml:lang="en" xmlns="http://www.aise.ics.
    saitama-u.ac.jp/qsl" xmlns:xs="http://www.w3.org/2001/
    XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
    instance">
3  <System>
4    <Security/>
5    <Environment>
6      <Server id="server001" purpose="registering"/>
7      <Server id="server002" purpose="submitting"/>...
8      <Gateway id="ip001" ref="device001" ip="10.0.5.1"/>
9      <Interface id="interface001" ref="device003"/>
10     <Device id="device001" type="PC" memory="8G" cpu="Intel
         Core i5" ref="soft001"/>
11     <Device id="device002" type="access point" encryption="
         WPA2-PSK" max_connection="50"/>
12     <Device id="device003" type="usb" edition="3.0" capacity="
         8G"/>...
13     <Software id="soft001" ref="server002" purpose="submitting
         " role="respondent"><Solution type="browser" name="
         chrome"/></Software>
14     <Software id="soft002" ref="server002" purpose="submitting
         " role="respondent"><Solution type="database" name="
         DB2" edition="8.1.6" data="participant"/></Software>
15   </Environment>
16   <Function>
17     <Phase>
```

```
18        <SettingUp>
19          <Import id="func001" role="questioner" scope="paper and
                setting" format="qsl"/>
20        </SettingUp>
21        <Distributing>
22          <Distribute id="func002" role="questioner" channel="usb
                "/>
23        </Distributing>
24        <Registering>
25          <Authenticate id="func003" role="respondent" method="
                token"/>
26        </Registering>
27        <Submitting>
28          <Ping id="func004" role="monitor" scope="ip"/>
29          <Observe id="func005" role="monitor" scope="state"/>
30          <Reply id="func006" role="respondent"/>
31          <Submit id="func007" role="respondent"/>
32          <Stop id="func008" role="respondent" method="interval"
                frequency="1" auto_save="yes"/>
33        </Submitting>
34        <Collecting>
35          <Collect id="func009" channel="usb"/>
36          <Integrate id="func010" role="questioner" scope="
                response" format="zip"/>
37        </Collecting>
38        <Marking>
39          <Mark id="func011" role="marker" scope="response"/>
40        </Marking>
41      </Phase>
42    </Function>
43  </System>
44 </QSL>
```

The above-mentioned snippet specification illustrates an offline e-testing system specification written by QSL. QSL also can be used to specify the corresponding service. In addition, the data are specified by QSL. As the data format, QSL is used to support the implementation of this system. Due to place constraints, the complete specifications please refer to Appendix A.

Chapter 4

Evaluation about Description Power of QSL

In order to approve the descriptive power of QSL that means QSL can be used to specify various e-questionnaire, e-testing, and e-voting systems, we evaluated QSL about the descriptive power from following four aspects: system, service, data format, and comparative study.

The system aspect is aimed to evaluate specifications by QSL for e-questionnaire, e-testing, and e-voting systems. The service aspect is aimed to evaluate specifications by QSL for e-questionnaire, e-testing, and e-voting on the systems. The aspect of data format is aimed to evaluate specifications by QSL for portable data. The evaluation steps of the three aspects are as follows: to investigate the target systems and the services, to enumerate requirements of the target systems, services, or results of the services, to summarize enumerated requirements, to specify the requirements by QSL, and to evaluate whether QSL provides enough tags to specify the requirements or not.

The aspect of comparative study is aimed to compare with the related works to evaluate the coverage of QSL for e-questionnaire, e-testing, e-voting, and the corresponding systems. The evaluation steps are to investigate existing specification language for e-questionnaire, e-testing, and e-voting, and enumerate their requirements, and compare with QSL to evaluate the coverage of QSL.

4.1 Specifications for E-Questionnaire, E-Testing, and E-Voting Systems

In order to verify that QSL can be used to specify various e-questionnaire, e-testing, and e-voting systems, at first, we investigated 25 e-questionnaire systems [1, 2, 5, 10, 11, 12, 14, 15, 16, 17, 22, 23, 31, 35, 36, 37, 38, 44, 49, 51, 52, 55, 56, 62], 22 e-testing systems [1, 10, 12, 17, 19, 20, 30, 33, 35, 38, 41, 42, 44, 46, 49, 50, 52, 58, 59, 62, 66, 74], and 23 e-voting systems [1, 10, 12, 17, 21, 25, 28, 35, 38, 44, 45, 47, 49, 52, 54, 57, 58, 61, 62, 67, 68, 71, 75], which are providing services for a lot of users in many companies, academies, and countries, for a long time. There are 11 systems providing e-questionnaire, e-testing, and e-voting services. We

enumerated requirements of each system. The 25 e-questionnaire systems have 60 requirements, the 22 e-testing systems have 70 requirements, and the 23 e-voting systems have 77 requirements. In addition, these systems have some common requirements for e-questionnaire, e-testing, and e-voting systems.

We used QSL to specify the summarized requirements by category of common requirements for three kinds of the systems, requirements for e-questionnaire systems, requirements for e-testing systems, and e-voting systems.

The evaluation result of the specification for e-questionnaire, e-testing, and e-voting systems is represented in Table 4.1. The columns list the number of the summarized requirements and the number of the requirements QSL can cover.

Table 4.1: Evaluation result of specification for e-questionnaire, e-testing, and e-voting systems

Category	**No. of requirement**	**No. of QSL coverage**
common	48	45
e-questionnaire systems	12	12
e-testing systems	22	22
e-voting systems	29	29

Current QSL is enough to specify 57 requirements for e-questionnaire, 67 requirements for e-testing, and 74 requirements for e-testing. The rest of the requirements that QSL cannot cover are listed below.

- **Error recovery**: the submitting server of e-questionnaire, e-testing, and e-voting systems shall run a self-check before a resuming is possible. In case of irreversible problems the server shall prevent a resuming of the submitting phase.
- **Response database encryption**: e-questionnaire, e-testing, and e-voting systems shall encrypt the response database.
- **Retrieve question**: e-questionnaire, e-testing, and e-voting systems shall be capable of finding and getting back questions before submitting the responses.

From the result, QSL can be used to specify more than 95% requirements and is easy to extend to specify the rest of the requirements caused by well-formed structure. The above-listed requirements are the common requirements that will be added and defined in 300-series schemas. In general, it proves that QSL can specify various e-questionnaire, e-testing, and e-voting systems. In the other words, QSL has the descriptive power for e-questionnaire, e-testing, and e-voting systems.

4.2 Specifications for E-Questionnaire, E-Testing, and E-Voting on the Systems

In order to evaluate that QSL can be used to specify various e-questionnaire, e-testing, and e-voting on the systems, we respectively chose 10 representative e-questionnaires on the Statistic Japan [60], 10 e-testing on National Education Examination Authority (NEEA) [39], and e-voting based on the Europe and US standards [70] launched on 9 popular systems [1, 10, 17, 35, 38, 49, 52, 58, 62] to provide e-questionnaire, e-testing, and e-voting services in the world. We enumerated the requirements for each service. We summarized the enumerated requirements. The 10 e-questionnaire have 65 requirements, the 10 e-testing have 55 requirements, and the 10 e-testing have 48 requirements. We also found these three kinds of e-services have in common such as the phases and paper structure.

We used QSL to specify the summarized requirements by category of common requirements for three kinds of the services, requirements for e-questionnaire, requirements for e-testing, and e-voting. The evaluation result of the specification for e-questionnaire, e-testing, and e-voting is shown in Table 4.2. The columns list the number of the summarized requirements and the number of the requirements QSL can cover.

Table 4.2: Evaluation result of specification for e-questionnaire, e-testing, and e-voting on the systems

Category	**No. of requirement**	**No. of QSL coverage**
common	40	38
e-questionnaire	25	24
e-testing	15	15
e-voting	8	8

Current QSL can specify 62 requirements for e-questionnaire, 53 requirements for e-testing, and 46 requirements for e-testing. The rest of the requirements that QSL cannot cover are listed below.

- **CSS style**: e-questionnaire, e-testing, and e-voting systems shall provide css style design for the paper design.
- **Dynamic multi-tier lookup table**: e-questionnaire system shall provide a text box to represent hierarchies of data likes parent to child.
- **Theme library**: e-questionnaire, e-testing, and e-voting systems shall provide a theme library for design the paper.

In addition, the reasons of the rest requirements QSL cannot cover, on the one hand is not necessary requirements for QSL because these requirements concern paper design but QSL focus on the functions of the services to support all the

phases of services so that it does not need to consider, such as "css style" and "theme library;" on the other hand is easy to extend QSL to contain it, such as the "dynamic multi-tier lookup table."

From the result, QSL can be used to specify more than 95.4% requirements and is easy to extend to specify the rest of the requirements caused by well-formed structure. The requirements named "dynamic multi-tier lookup table" will be added and defined in 300-series schemas. In summary, It proves that QSL can specify various e-questionnaire, e-testing, and e-voting. In the other words, QSL has the descriptive power for e-questionnaire, e-testing, and e-voting on the systems.

4.3 Specifications for Portable Data

In order to verify that QSL can be used as format of portable data of e-questionnaire, e-testing, and e-voting systems, that means to check whether QSL can provide enough tags and notations to describe the results of e-questionnaire, e-testing, and e-voting, we referred to the method listed up previously. We launched the e-questionnaire, e-testing, and e-voting, and imitated the respondents to answer so that obtained the results of the e-questionnaire, e-testing, and e-voting. We enumerated the requirements of the results from each service, and summarized the enumerated requirements. The 10 e-questionnaire have 13 requirements, the 10 e-testing have 16 requirements, and the 10 e-testing have 11 requirements for specifications of data. We also found some requirements of data have in common.

In addition, we used QSL to specify the results. Current QSL can specify 13 requirements for e-questionnaire, 16 requirements for e-testing, and 11 requirements for e-testing. Table 4.3 illustrates the evaluation result of the specification for portable data about e-questionnaire, e-testing, and e-voting systems. The columns list the number of the summarized requirements and the number of the requirements QSL can cover.

Table 4.3: Evaluation result of specification for portable data

Category	**No. of requirement**	**No. of QSL coverage**
common	6	6
e-questionnaire systems	7	7
e-testing systems	10	10
e-voting systems	5	5

From the result, QSL can specify the whole requirements. In conclusion, it proves that QSL can specify various data of the e-questionnaire, e-testing, and e-voting. In the other words, QSL has descriptive power for portable data.

4.4 Comparison with Related Works

In order to verify that the descriptive power of QSL can cover various e-questionnaire, e-testing, and e-voting systems, we compared with existing specification languages with different motivations for specifying e-questionnaire, e-testing, and e-voting systems, respectively. All the existing languages are based on XML.

At first, Bethke [8] proposed a questionnaire specification language in Simple Survey Systems (SSS) to provide a straight-forward way to present questionnaires to untrained users for survey research. The language can be use to specify structure, content, and a part of question types of questionnaires, as well as two kinds of dynamic logic types [7]. SSS focuses on presenting what a simple questionnaire has. Secondly, IMS Question and Test Interoperability Specification (QTI) [29] is aimed at providing a standard format for the representation of assessment content and results, supporting the exchange of this material between authoring and delivery systems, repositories and other learning management systems. QTI focuses on presenting what a test paper and a result report have, and how to process response and operation for data exchange. Thirdly, OASIS Election Markup Language (EML) [40] is a standardized XML language for 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. EML focuses on describing what and how the data interchange in a secure election process in details.

Compared with the above-mentioned languages, QSL is aimed at providing a unique and common method for the communications of the stakeholders of e-questionnaire, e-testing, and e-voting systems. Because of this motivation, QSL should have the descriptive power to specify what the e-questionnaire, e-testing, and e-voting systems have. In other words, QSL is able to specify the requirements for various e-questionnaire, e-testing, and e-voting systems, which are listed in Table 4.4. This table also shows the requirements what QTI and EML can specify. Because the terminologies of QSL and the related works are different, the requirements are described according to QSL terminology.

Fig. 4.1 illustrates a straight-forward relation overview of descriptive power of QSL, SSS, QTI, and EML. From this figure, QSL can specify all the requirements to satisfy the motivation.As we found that QSL can specify most of the requirements except them in region 8 and region 9.

Until now QSL is not considering e-voting for e-learning services, and the css style. Hence, the items in region 8 are not the necessary requirements of QSL. Moreover, the identifier is used to refer to the EML-based program for connect with the EML schemas. The bureaucracy information such as some office hours are not necessary for QSL. QSL focuses on what the systems have, what the phases have, so the requirements how to process and the detailed information having no concerns with systems and phases are also not necessary requirements for QSL. The above-mentioned requirements in region 8 and region 9 are the optional requirements that QSL can be slightly and easily extended.

From the result, QSL is a better tool than SSS, QTI, and EML. Because QSL has enough coverage for specifying various e-questionnaire, e-testing, and e-voting

Table 4.4: A list of the requirements with related works.

Region	Requirements
#1	paper: section, question, answer, etc.
	question type: single choice, multiple choices
#2	question type: multiple-row box, single-row text
#3	logic: simple skipping, compound skipping
#4	setting: media, weighting, etc.
	data: sample, score, comment, etc.
	question type: drag, connect the points, slide, etc.
	logic: randomization
#5	question type: formula, drop-down list, matrix, etc.
	setting: ordering, numbering, etc.
	data: respondent information, result
#6	participants: candidate, executor
	setting: multi-language, launch/finish time. etc.
	phase: setting up, distributing, registering, submitting, collecting, analyzing, counting, auditing
	security: authentication, anonymity, channel, etc.
	environment: interface and it communicates with security
#7	participant: sponsor, executor (marker), auditor
	logic: compound skipping, piping, extraction
	setting: spelling checking, automatic lookup, etc.
	phase: marking
	environment: server, database, device, software, etc.
	function: export, import, monitor, etc.
#8	function: tester feedback sample correctness
	setting: css, attempt times for incorrect answer, etc.
#9	data: bureaucracy information, etc.
	environment: processing unit, identifier with election, etc.

systems, as well the stakeholders can only learn QSL to deal with the three kinds of systems. After all, when the stakeholders use QTI, SSS, or EML, they have to learn three languages to deal with three systems. In addition, There are existing 11 systems providing e-questionnaire, e-testing, and e-voting services we mentioned and investigated above. It is necessary to provide specifications for e-questionnaire, e-testing, and e-voting services, as well as the corresponding systems. QSL can satisfy the demand in tend and can be used to specify these three kinds of services and the corresponding systems.

4.5 Summary

In this chapter, we present the evaluation about the descriptive power of QSL from four methods and the results to prove that QSL can specify various e-questionnaire, e-testing, e-voting systems, as well as the data and the corresponding systems. In

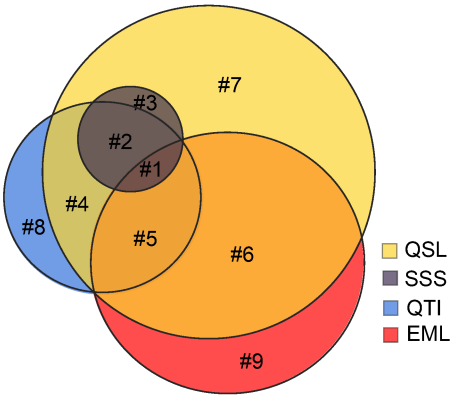

Figure 4.1: Descriptive power of QSL with related works.

addition, QSL is a better tool because of the coverage of specifications for e-questionnaire, e-testing, and e-voting systems. In general, QSL has the generality and availability.

Chapter 5

Applications for QSL

In this chapter, it shows applications for two aspects: one aspect is about an alternative usage of QSL in real applications we implemented for data portability. Another aspect is about expectable applications for QSL we are ready to implement.

5.1 A Format of Data Portability

Data Portability [6] is the ability for users to reuse their own data between interoperable applications, and is getting treated as an important characteristic of web and cloud services in recent years. It is said that improving data portability activates competition among websites, and provide a better environment for web users. In addition, the right to data portability is accepted by General Data Protection Regulation [18] in 2016. Data portability is very important in our modern society.

According to e-questionnaire, e-testing, and e-voting systems are indispensable for our society and have many users. The lack of data portability causes several problems. Firstly, if the system does not have enough data portability, for example, a user wants to move data among systems or change their service provider. Different systems commonly have proprietary data formats, templates, and related parameters that tend to lock users into specific systems, after all, the specific systems have many users. Secondly, as a personal problem, users cannot freely manage their own data. Last but not the less, they will have to give up a lot of data when they change systems.

Therefore, it is necessary to provide a format of portable data for implementing data portability of e-questionnaire, e-testing, and e-voting systems. There are two real applications we implemented that QSL is used as a unified format of portable Data.

5.1.1 ENQUETE-BAISE: a General-Purpose E-Questionnaire Server for Ubiquitous Questionnaire

ENQUETE-BAISE [9] is a general-purpose e-questionnaire server developing for a ubiquitous questionnaire that can be used as a readymade e-questionnaire server component in various web service systems as well as an alone e-questionnaire server with general-purpose for various questionnaires. It can also be used as an e-testing server and an e-voting server with general-purpose by restricting its general functions and strengthening its security functions [27]. ENQUETE-BAISE [3] has been used to support students at Saitama University to access, login, and use it to do e-questionnaire, e-testing, and e-voting since 2007.

ENQUETE-BAISE lacks data portability because it did not have a format of portable data. QSL is used as a data format for improving data portability of ENQUETE-BAISE [32]. It helps to conduct an e-questionnaire on different e-questionnaire systems. The data of paper, setting, responses, and result have been specified by QSL.

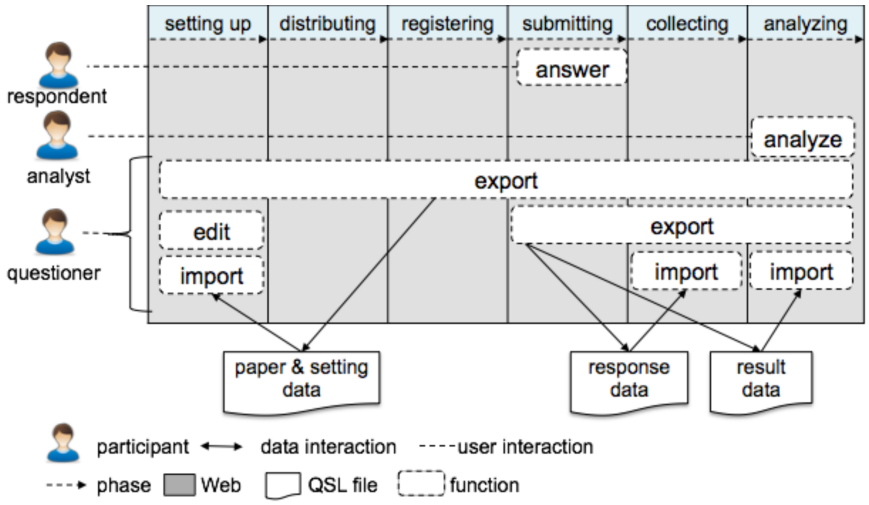

Figure 5.1: Overview of ENQUETE-BAISE for portable data.

Fig. 5.1 shows overview of ENQUETE-BAISE for portable data. We implemented and improved ENQUETE-BAISE for data portability. All the data are exchanged in a QSL-format requirement specification file. A questioner can edit a paper and set up settings for the paper. The paper data and setting data are saved in the database in real-time. The questioner also can import a QSL-format requirement specification recording paper data and setting data during setting up phase. The questioner can export paper data and setting data by QSL format during the whole phases. The respondents answer questions, and the response data are saved in database during submitting phase. The questioner can export and download the submitted response data. After collecting the responses, all the response data are all integrated and send to the analyst to analyze and get the result. ENQUETE-BAISE can automatically analyze and get the result. The result

data is saved in the database and the questioner can export the result data. Sometimes, the questioner can import response data to analyze by ENQUETE-BAISE, and import the result data to record in the database.

5.1.2 A General-Purpose Offline E-Testing Environment

We implemented a general-purpose offline e-testing environment [72] based on QSL to provide users with offline e-testing service to execute various offline e-testing. This environment has been applied to execute the final test of Discrete Mathematics in the University of Japan since 2015 [24].

Considering it is difficult for teachers to manage test data when to execute offline e-testing in classrooms, QSL is used as a data format to solve the difficulty about management such that the environment provides users with offline e-testing service to execute various offline e-testing. The data of exam paper, participant, responses, and result have been developed by QSL, and the QSL-format data has been used directly. Moreover, we used QSL to define all the requirements of this system in the phase of pre-development.

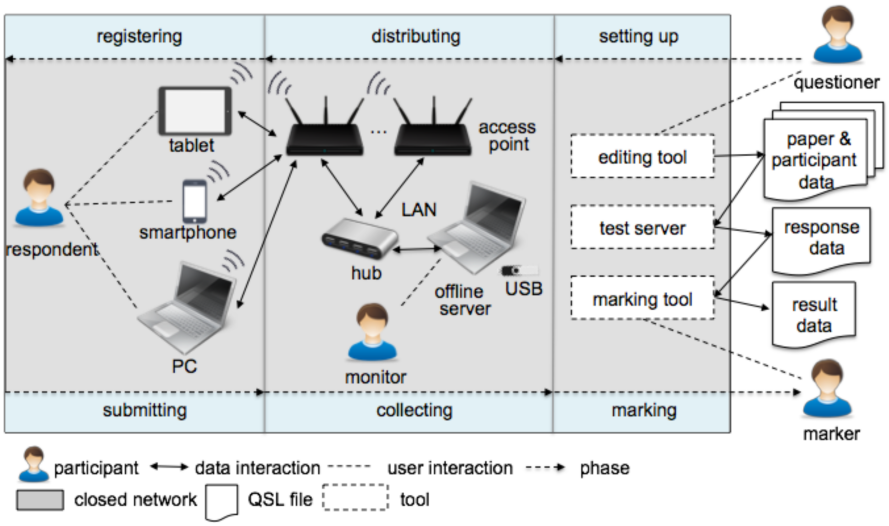

Figure 5.2: Overview of general-purpose offline e-testing environment for portable data.

The overview of a general-purpose offline e-testing environment is illustrated in Fig. 5.2. An editing tool is set into the USB flash memory. A questioner uses the editing tool to specify all participants by QSL. After that, the editing tool will generate the admission ticket for each respondent automatically. The questioner uses the editing tool to prepare exam paper with complex question types and logic, which files are specified by QSL. An offline test server distributes exam paper to respondents through wireless LAN. The environment uses the access point to support to test a large number of the respondents easily and conveniently. The respondents confirm the exam paper from the offline server. During the test, the monitor monitors the connection states of all the respondents through the

offline server for avoiding online cheating activities, and the closed network will also block accesses from outside to connect to the offline server. After submitting, all the responses will be collected and integrated by the offline server as QSL-format files. The offline server distributes and collects offline e-testing. Questioner only needs to stick the USB flash memory into a PC, the PC can be the offline server to execute offline e-testing. The marking tool is also set into the USB flash memory. A marker uses the marking tool to mark the collected responses and give a result of the test as a QSL-format file. An analyst uses the marking tool to analyze responses and get the test result automatically. The functions of the tools and phases, as well as the environments, are defined by QSL.

5.2 Expectable Applications for QSL

Because QSL is XML-based language, differ with the natural language, it is machine-readable. There are two expectable applications for QSL:

- **QSL verifier** to check whether the requirements in specifications are enough or not, and check whether the specification conforms to the standards, laws, and regulations.
- **System generator** to convert from QSL-format requirement specifications to generate e-questionnaire, e-testing, and e-voting systems easily and automatically.

We summarized the relationships of the requirements, so it is possible to implement a QSL Verifier is an application to check whether the requirements in specifications are enough or not, and check whether the specification conforms to the standards, laws, and regulations that can help stakeholders simplify and strictly verify their works. Another application is a system generator to convert from QSL-format requirement specifications easily and automatically to generate e-questionnaire, e-testing, and e-voting systems that are used to streamline the stakeholders' procedures. According to the above two applications, it also proves that QSL can be used to solve communication problems. Because the stakeholders do not need to check the requirements, and it helps to jump the programming procedure.

Fig. 5.3 shows the expectable applications for QSL, which are QSL verifier and system generator. The user (usually sponsor and executor) discuss and use QSL structure editor to input values to get a service requirement specification. Based on the service requirement specification, the supporter can design a system requirement specification of by this editor. This editor can help users to write specification easily. The requirement specification is as the input data into a QSL verifier, and the verifier output the result of the verification. If the result is yes, then output the reliable requirement specification and store the output data into QSL database. If there are errors or mistakes, then output the feedback to the user, and the user can improve the specification by checking and according to the feedback. If the users want to reuse QSL document, it is necessary to provide database. And the

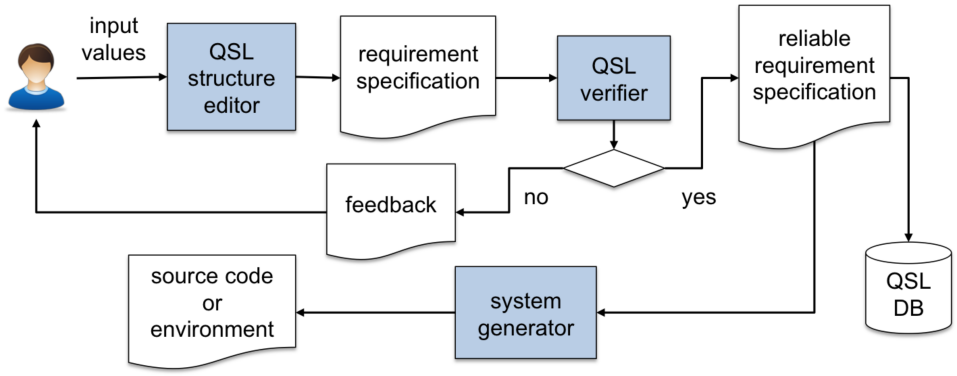

Figure 5.3: QSL verifier and system generator.

editor and verifier can also use the data in QSL database. Because QSL-format specifications are machine-readable data, the system generator inputs the reliable requirement specification, and can generate some encapsulated code. The code through the processor and run can get a system to do e-questionnaire, e-testing, and e-voting.

5.3 Summary

QSL can be used as a unified data format for portable data of e-questionnaire, e-testing, and e-voting systems. There are two real applications to prove its availability for data portability of QSL.

Expectable applications for QSL can be used to simplify the stakeholders' works and streamline their procedures, and even solve the communication problems.

Chapter 6

Conclusions

6.1 Contributions

This thesis proposes the first specification language, named "QSL," 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 precondition of automatically generating e-questionnaire, e-testing, and e-voting systems. QSL can be used to provide to the corresponding stakeholders a desirably exhaustive requirement list and the relationships among the requirements so that they can easily choose requirements and complete the requirement specifications.

To propose QSL, firstly, we collected, analyzed, and summarized exhaustive requirements for systems and services of e-questionnaire, e-testing, and e-voting, which are extremely similar. There are 181 requirements for e-questionnaire systems, 169 requirements for e-testing systems, and 168 requirements for e-voting systems. We summarized 121 common requirements for e-questionnaire, e-testing, and e-voting systems. The exhaustive requirements are the fundament of QSL and based on the requirements, the stakeholders can choose requirements and complete the requirement specifications easily.

Secondly, according to the investigations, we proposed QSL as a unified method to provide terminology and specifications of e-questionnaire, e-testing, and e-voting systems with a standardized, consistent, and exhaustive requirement list to solve the communication problems among the stakeholders.

Thirdly, we evaluated the proposed QSL about description power to ensure its completeness manifesting in specifying various e-questionnaire, e-testing, and e-voting systems. Current QSL can cover more than 95% system requirements and 95.4% service requirements, and provides enough notations to describe the requirements for data portability, and can cover the existing specifications compared with the related works. The reasons of the rest requirements QSL cannot cover, on the one hand, they are not necessary requirements for QSL according to the motivation, and on the other hand, they are easy to extend QSL to cover them.

Lastly, we proposed an alternative usage of QSL for data portability and implemented two real applications for QSL. QSL is used as a data format for data portability for a general-purpose e-questionnaire server to help conduct an

e-questionnaire on different e-questionnaire systems, and for an offline e-testing environment to provide users with offline e-testing service to execute various offline e-testing. The applications can prove QSL is practical and available to be used to provide data format for data portability.

6.2 Future works

The ultimate goal of this research is to support all kinds of stakeholders to easily communicate with each other by QSL.

It is needed to provide a standardized, consistent, and exhaustive requirement list for all kinds of stakeholders. We have designed and developed QSL (version 3.1) to provide an exhaustive requirement list oriented to different kinds of stakeholders and the relationships among the requirements. In addition, current QSL can cover more than 95% existing requirements we investigated. We continue improving QSL though case studies in order to provide a complete requirement list.

In order to easily write requirement specifications, a QSL structure editor can provide the stakeholders with an intuitive interface to easily choose the desirable requirements from an exhaustive requirement list and generate the requirement specification. Until now, we use an XML editor to convert from QSL schemas to QSL templates and fill the values in the templates to get the requirement specifications. We will implement a QSL structure editor to easily write QSL-format requirement specifications.

In the future, a series of supporting tools for QSL such as QSL verifier and system generator should have been developed. It can be used to simplify the stakeholders' works and streamline their procedures, and even solve the communication problems.

Publications

Refereed papers published in journals or books (first author)

- Yuan Zhou, Hongbiao Gao, and Jingde Cheng: QSL: A Specification Language for E-Questionnaire, E-Testing, and E-Voting Systems, in J. J. Park, H. Jin, Y. Jeong and M. K. Khan (Eds.), "Advanced Multimedia and Ubiquitous Engineering-FutureTech & MUE," Lecture Notes in Electrical Engineering, Vol. 393, pp. 255-261, Springer, Singapore, August 2016.
- Yuan Zhou, Hongbiao Gao, and Jingde Cheng: QSL: An Extension of QSL for E-voting Systems, in J. J. Park, Y. Pan, G. Yi and V. Loia (Eds.), "Advances in Computer Science and Ubiquitous Computing - CSA-CUTE2016," Lecture Notes in Electrical Engineering, Vol. 421, pp. 87-96, Springer, Singapore, November 2016.
- Yuan Zhou, Yuichi Goto, Jingde Cheng: QSL: A Specification Language for E-Questionnaire, E-Testing, and E-Voting Systems, IEICE Transactions on Information and Systems. (Submitted)

Refereed papers published in international conference proceedings (first author)

- Yuan Zhou, Yuichi Goto, Jingde Cheng: QSL: A Specification Language for E-questionnaire Systems, Proceedings of the 5th IEEE International Conference on Software Engineering and Service Science (ICSESS 2014), pp. 224-230, Beijing, China, IEEE press, June 2014.
- Yuan Zhou, Daisuke Matsuura, Yuichi Goto, and Jingde Cheng: Evaluation about the Descriptive Power of QSL: A Specification Language for E-Questionnaire, E-Testing, and E-Voting Systems, Proc. of 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communication, Cloud & Big Data Computing, Internet of People and Smart City Innovations (SmartWorld/ UIC/ATC/ ScalCom/ CBDCom/ IOP/ SCI 2018), pp. 198-203, Guangzhou, China, IEEE-CS, Oct. 2018.

Refereed papers published in journals or books (co-author)

- Zhe Wang, Yuan Zhou, Bo Wang, Yuichi Goto, and Jingde Cheng: An Extension of QSL for E-testing and Its Application in an Offline E-testing Environment, in J. J. Park, H. Chao, H. Arabnia, and N. Y. Yen

(Eds.), "Advanced Multimedia and Ubiquitous Engineering - Future Information Technology," Lecture Notes in Electrical Engineering, Vol. 352, pp. 7-14, Springer, Heidelberg, May 2015.

Refereed papers published in international conference proceedings (co-author)

- Hongbiao Gao, Zhe Wang, Yuan Zhou, and Jingde Cheng: Development of a General-Purpose Offline E-Testing Environment, Proceedings of the 12th International Conference on Computational Intelligence and Security (CIS 2016), pp. 603-607, Wuxi, China, IEEE Computer Society Press, December 2016.

References

[1] AddPoll, Best Online Survey and Poll Software, http://www.addpoll.com

[2] Ai Diao Yan (in Chinese), Community Leading Internet Research, http://www.idiaoyan.com

[3] Advanced Information Systems Engineering Laboratory, Department of Information and Computer Sciences, Saitama University, ENQUETE-BAISE: A General-Purpose E-Questionnaire Server for Ubiquitous Questionnaire, http://www.aise.ics.saitama-u.ac.jp/enquete/index.html

[4] Advanced Information Systems Engineering Laboratory, Department of Information and Computer Sciences, Saitama University, Questionnaire Specification Language (QSL) Specification (version 3.1), http://www.aise.ics.saitama-u.ac.jp/QSL/QSLSpecification_v3.1.pdf

[5] AskForm (in Chinese), Professional Enterprise Testing Platform, http://www.askform.cn

[6] Paul R. BENSON: Data Portability: It Is About the Data; the Quality of the Data. MIT Information Quality Industry Symposium, pp. 614-618, July 2009.

[7] Albert D. BETHKE: Representing Procedural Logic in XML, Journal of Software, Vol. 3, No. 2, pp. 33-40, 2008.

[8] Albert D. BETHKE: Using XML As a Questionnaire Specification Language, IEEE Proceedings of SoutheastCon 07, Richmond, pp. 127–131, 2007.

[9] Jingde CHENG, Yuichi GOTO, Masato KOIDE, Keigo NAGAHAMA, Masami SOMEYA, Yusuke UTSUMI, and Ayaka SHIONOIRI: ENQUETE-BAISE: A General-Purpose E-Questionnaire Server for Ubiquitous Questionnaire, In Proceedings of IEEE Asia-Pacific Services Computing Conference, IEEE-CS, Tsukuba, Japan, pp. 187-194, December 2007.

[10] Constant Contact, Effective Email Marketing, http://www.constantcontact.com

[11] CubeQuery (in Japanese), A Free Questionnaire Form, http://cubequery.jp

[12] Diao Cha Quan (in Chinese), Enterprise Online Survey and Management Platform, http://www.diaochaquan.cn

[13] Sannella, DONALD and Martin WIRSING: Specification Languages, *Algebraic Foundations of Systems Specification*, Springer, Berlin, Heidelberg, pp. 243-272, 1999.

[14] Dounano (in Japanese), http://www.dounano.jp, accessed Aug. 25. 2018.

[15] EnableQ, http://enableq.chinaedu.net/System/Login.php

[16] Enq-maker (in Japanese), https://enq-maker.softonic.jp/web

[17] eSurv, Free Survey Maker, http://eSurv.org

[18] European Commission, Policies, Information and Service, Data Protection, https://ec.europa.eu/info/law/law-topic/data-protection_en

[19] ExamSoft, Exam Software: Formative Assessments and Summative Assessments, https://learn.examsoft.com

[20] Examcoo (in Chinese), A Free Online Platform for Exam and Home Work, http://www.examcoo.com

[21] FC2 Vote (in Japanese), http://vote.fc2.com

[22] Fluidsurveys, Fluidsurveys Has Joined the SurveyMonkey Family, http://fluidsurveys.com

[23] FormSite, Online Form Builder, http://www.formsite.com

[24] Hongbiao GAO, Zhe WANG, Yuan ZHOU, and Jingde CHENG: Development of A General-Purpose Offline E-Testing Environment, In Proceedings of the 12th International Conference on Computational Intelligence and Security (CIS 2016), IEEE, Wuxi, China, pp. 603-607, 2016.

[25] Google Form, Forms: Free Online Surveys for Person Use, http://www.google.cn/intl/zh_cn/forms/about/

[26] Thomas R. GRUBER, Thomas R: A Translation Approach to Portable Ontology Specifications, *Knowledge acquisition*, Vol.5, No. 2, pp. 199-220, 1993.

[27] Yuichi GOTO and Jingde CHENG: Information Assurance, Privacy, and Security in Ubiquitous Questionnaire, In Proceedings of the 4th International Conference on Frontier of Computer Science and Technology, IEEE-CS, Shanghai, China, pp. 619-624, 2009.

[28] Helio, https://vote.heliosvoting.org

[29] IMS Question and Test Interoperability Specification Overview, IMS Global Learning Consortium, http://www.imsglobal.org/question/

[30] Ischool Random Testing System (ver. 3.7.1), http://down.chinaz.com/soft/26226.htm, accessed Aug. 25. 2018.

[31] Key Survey, Online Survey Software & Questionnaire Tool, and surveys maker, https://www.keysurvey.com

[32] Yohei KAMATA and Yuichi GOTO: Improvement of Data Portability of ENQUETE-BAISE: A General-Purpose E-Questionnaire Server for Ubiquitous Questionnaire, In Proceedings of 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communication, Cloud & Big Data Computing, Internet of People and Smart City Innovations (SmartWorld/ UIC/ATC/ ScalCom/ CBDCom/ IOP/ SCI 2018), pp. 174-179, Guangzhou, China, IEEE-CS, October 2018.

[33] LoveKao (in Chinese), http://down.chinaz.com/soft/29038.htm

[34] Ulle MADISE and Tarvi MARTENS: E-Voting in Estonia 2005. The First Practice of Country-Wide Binding Internet Voting in the World, Jr.: *Electronic Voting, No. 86*, 2006.

[35] Mobo Survey, http://www.mobosurvey.com

[36] Moodle, Open Source Learning Platform, http://www.moodle.org

[37] My3Q (in Chinese), http://www.my3q.com

[38] MySurveyLab, Best Survey Tool. Questionnaire, Tests and Forms, https://www.mysurveylab.com

[39] National Education Examinations Authority, http://www.neea.edu.cn

[40] OASIS Election Markup language (EML) Specification Version 7.0, http://docs.oasis-open.org/election/eml/v7.0/cs01/eml-v7.0-cs01.html

[41] OCR, Oxford Cambridge and RSA, OCR - Awarding Body for A Levels, GCSEs, Cambridge Nationals, Cambridge Technicals and Other Qualifications, https://www.ocr.org.uk

[42] oExam (in Chinese), http://www.orivon.com

[43] Ontology Development 101: A Guide to Creating Your First Ontology, https://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

[44] OQSS, Web Survey & Form, http://www.oqss.com

[45] Opinion Stage, Create Poll, Quizzes, Lists and More, https://www.opinionstage.com

[46] PHPEMS Online Testing System, http://www.phpems.net

[47] Poll Every Where, Live Interactive Audience Participation, https://www.polleverywhere.com

[48] POLYAS, Secure Online Voting with POLYAS, https://www.polyas.com

[49] ProProfs, Knowledge Management Software, http://www.proprofs.com

[50] Qi Bo Testing System (in Chinese), http://down.chinaz.com/soft/29998.htm

[51] Qualtrics, The Leading Research & Experience Software, http://www.qualtrics.com

[52] QuestionPro, Customer Satisfaction Surveys Online Web Questionnaire, http://www.questionpro.com

[53] Nick ROZANSKI and Ein WOODS: Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives, *Addison-Wesley*, 2012.

[54] Simple Voting, https://www.simplyvoting.com

[55] Smart Survey, Online Survey Software & Questionnaire Tool, http://www.smartsurvey.co.uk

[56] Smaster (in Japanese), http://www.smaster.jp

[57] Snappy Poll, https://www.snappypoll.com

[58] So Jump (in Chinese), Questionnaire and Online Exam, http://www.sojump.com

[59] Sogo Survey, Free Online Survey Software, http://www.sogosurvey.com

[60] Statistic Bureau, Ministry of Internal Affairs and Communications, http://www.stat.go.jp

[61] Stone Poll (in Chinese), WeChat Voting, http://www.stonepoll.com

[62] SurveyMonkey, http://www.surveymonkey.com

[63] Survey Moz, Create Online Survey for Free, http://www.surveymoz.com

[64] Surveyi, Online Survey Design, Publishing and Analysis Tools, http://www.surveyi.com

[65] Tian Hui Diao Yan Bao (in Chinese), http://www.diaoyanbao.com

[66] Tomexam Online Testing System (in Chinese), http://www.tomexam.com

[67] Tou Piao Wang (in Chinese), www.toutoupiao.com

[68] Vizzual Forms, Suevey and HTML Form Builder. Create Web Questionnaires for Free, http://www.vizzualforms.com

[69] Eric van der VLIST: XML Schema, *O' Reilly*, New York, 2002.

[70] Melanie VOLKAMER: Evaluation of Electronic Voting: Requirements and Evaluation Procedures to Support Responsible Election Authorities, *Lecture Notes in Business Information Processing, Vol. 30*, Springer, 2009.

[71] Votenet, http://www.votenet.com

[72] Zhe WANG, Yuan ZHOU, Bo WANG, Yuichi GOTO, and Jingde CHENG: An Extension of QSL for E-Testing and Its Application in An Offline E-Testing Environment, In J. J. Park, et al. (Eds.), Advanced Multimedia and Ubiquitous Engineering, Lecture Notes in Electrical Engineering (LNEE), Springer, Vol. 352, pp. 7-14, 2015.

[73] W3C: Extensible Markup Language (XML) 1.0 (Fifth Edition), http://www.w3.org/TR/2008/REC-xml-20081126/

[74] Yong Dao Offline Testing (in Chinese), http://www.onlinedown.net/soft/49716.htm

[75] YouPoll, http://www.youpolls.com

[76] Yuan ZHOU, Hongbiao GAO, and Jingde CHENG: An Extension of QSL for E-Voting Systems, In J. J. Park, et al. (Eds.), Advances in Computer Science and Ubiquitous Computing, Lecture Notes in Electrical Engineering, Springer, Vol. 421, pp. 87-96, 2016.

[77] Yuan ZHOU, Hongbiao GAO, and Jingde CHENG: QSL: A Specification Language for E-Questionnaire, E-Testing, E-Voting Systems, In J. J. Park, et al. (Eds.), Advanced Multimedia and Ubiquitous Engineering, Lecture Notes in Electrical Engineering, Springer, Vol. 393, pp. 255-261, 2016.

[78] Yuan ZHOU, Daisuke MATSUURA, Yuichi GOTO, and Jingde CHENG: Evaluation About the Descriptive Power of QSL: A Specification Language for E-Questionnaire, E-Testing, and E-Voting Systems, In Proceedings of 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communication, Cloud & Big Data Computing, Internet of People and Smart City Innovations (SmartWorld/ UIC/ATC/ ScalCom/ CBDCom/ IOP/ SCI 2018), IEEE-CS, Guangzhou, China, pp. 198-203, Oct. 2018.

[79] Yuan ZHOU, Hongbiao GAO, and Jingde CHENG: QSL: A Apecification Language for E-Questionnaire Systems, In Proceedings of the 5th IEEE International Conference on Software Engineering and Service Science (ICSESS 2014), IEEE, Beijing, China, pp. 224-230, 2014.

Appendix A

QSL Specification (Version 3.1)

Specification URIs

- This version:
  http://www.aise.ics.saitama-u.ac.jp/QSL/QSLSpecification_v3.1.pdf
- Previous version:
  http://www.aise.ics.saitama-u.ac.jp/QSL/QSLSpecification_v3.0.pdf

QSL Members

- Y. Zhou (shuugen@aise.ics.saitama-u.ac.jp), D3
- Z. Wang, M.S. graduated in 2015
- D. Matsuura, B.S. graduated in 2016

Author and Editor

- Y. Zhou (shuugen@aise.ics.saitama-u.ac.jp), D3

QSL Members

- Y. Zhou (shuugen@aise.ics.saitama-u.ac.jp), D3
- Z. Wang, M.S. graduated in 2015
- D. Matsuura, B.S. graduated in 2016

Abstract

This document describes the background and purpose of Questionnaire Specification Language (QSL), presents the grammar of QSL in XML Schema including the whole elements, attributes, and the structure of it, and shows system template and service template for specifying systems and service of e-questionnaire, e-testing, and e-voting. We devote to continuously improve QSL better and better since 2013 till now.

A.1 Introduction

A.1.1 QSL Documentations

- **QSL Document/ Requirement Specification**: the specification using QSL to describe is written by the user who uses QSL apps or other XML editors.
- **QSL Specification**: the QSL manual to guide user how to use QSL to specify systems and services of e-questionnaire, e-testing, and e-voting.
- **Schema**: QSL grammar for supporting QSL. XML Schema defines these schemas. A schema is the structure of skeletons of requirements.
- **Skeleton of Requirements**: a complete list of requirements or a document of requirements without values. The stakeholders only need to fill in the values to complete the requirement specification.
- **Tag**: is used to describe a piece of data for requirements of e-questionnaire, e-testing, and e-voting systems.
- **Template**: is a set of skeletons of requirements.

A.1.2 Background

AISE members, to design a formalized method for various e-questionnaire, e-testing, and e-voting systems. The mission statement is, in part, to: "to propose a standard and formalized tool to solve the problems among the stakeholders of various e- questionnaire, e-testing, and e-voting systems ⋯"

The original objective in 2013 was to introduce a uniform and unique method to specify systems involved in the processes of e-questionnaire, e-testing, and e-voting, including data portability. The overall focus today provides a standard that is:

- Multinational: Our focus is to have standards that can be adopted globally.
- Flexible: Effective across the various questionnaires, testing, and polling regimes.
- Multilingual: Flexible enough to accommodate the various languages and dialects and vocabularies.
- Adaptable: Resilient enough to support questionnaires, examinations, and elections in both the private and public sectors.
- Secure: Able to secure the relevant data from ant attempt, as appropriate to the different requirements of various e-questionnaire, e-testing, and e-voting rules.

Questionnaire Specification Language (QSL) serves as a formalized specification for specifying various e-questionnaire, e-testing, and e-voting systems. QSL provides vocabulary and notation with a standardized, consistent, and exhaustive requirement list for specifying systems and services of e-questionnaire, e-testing, and e-voting. QSL is a format for portable data. QSL is the precondition to automatically generate e-questionnaire, e-testing, and e-voting. QSL is based on Extensible Markup Language (XML). The grammar of QSL is defined by XML Schema.

At present, QSL includes specifications for two templates and their data information:

- **System Template**: security requirements, environmental requirements including software and hardware, function requirements during phases of setting up, registering, distributing, submitting, collecting, analyzing, counting, marking, auditing.
- **Service Template**: requirements for paper sheet, settings for paper sheets during phases.
- **System Data**: participant fields for each role, paper fields, and response and result fields.
- **Service Data**: participant information, real responses by respondents, and counts, statistics, and results.

This document and its schemas and structures of two templates represent our best current efforts, knowledge and experience with e-questionnaire, e-testing, and e-voting systems since 2013. It is incumbent on users of this document to identify and requirement gaps, mistakes, inconsistencies or missing data and to propose corrections or enhancement to AISE.

A.1.3 Overview of the Document

To help establish the context for the specifications contained in the XML Schema that make up QSL, the first questionnaire specification language for e-questionnaire, e-testing, and e-voting systems. AISE also designed the structure of it. This structure identifies how to combine each element to obtain various e-questionnaire, e-testing, and e-voting systems. In this document, we present a lot of specifications as templates for references and reusability and describe how QSL can be used to standardize the data exchanged.

A.1.4 Changes in this Version

The changes from QSL v 3.0 that this new version introduces are as follows:

- Add much more schemas for common requirements of e-questionnaire, e-testing, and e-voting, which are extracted from 200-series schemas in version 3.0;
- Revise some attributes to let 32 requirements QSL can specify.

A.1.5 Advantages of using QSL

The question usually asked about why any particular standard should be used is "what advantages will it bring me? " In addition to supporting trusted e-questionnaire, e-testing, and e-voting systems, the benefits of adopting QSL are as follows:

Firstly, QSL can specify various systems and services of e-questionnaire, e-testing, and e-voting. It satisfies public demands to unify e-questionnaire, e-testing, and e-voting systems, and promotes the communications of stakeholders among these essentially similar systems. Secondly, QSL can be a unified format for portable data among different e-questionnaire, e-testing, and e-voting systems. It contributes to improve data portability, which right is very important for our modern society and accepted by General Data Protection Regulation since 2016 for providing conveniences for the stakeholders to reuse data. Thirdly, QSL can be the precondition to automatically generate e-questionnaire, e-testing, and e-voting systems. It is conducive to improving efficiency by automating and streamlining stakeholders' work procedures.

For sponsors:

- More choice of products and supporters;
- Clearly mind and unambiguous;
- Supports scalability, transparency, and data reusability;
- Provides basis for make clearly work division, and responsibility;

For executors:

- More choice of survey sheets;
- Clearly work division, and responsibility;
- Supports scalability, transparency, and data reusability;

For respondents:

- Supports trustworthiness of systems;
- Support security of e-questionnaire, e-testing, and e-voting;

For supporters:

- Easily communicate with sponsors and executors;
- Reduced development costs;
- Accommodates future changes more easily;
- Common core but allows local extension;

For evaluators:

- Easily check specifications.

A.1.6 How to use QSL

As a specification language, QSL has to meet various requirements of systems and services of e-questionnaire, e-testing, and e-voting. Therefore it may need to be tailored for specific scenarios and meet specific rules and practices.

First using the QSL Schemas (XSD files) and an xml-editor to translate it and to build an XML file, that is a QSL template. You can also use a requirement specification or a template, to change it or fill the values by your own mind but do not need to break up the QSL grammar definition. Secondly, because we have the marked numbers for each schema, you can select which kinds of systems and services you must to specify. Thirdly, you can use a QSL Structure Editor to select the role of your owns, choose the template, fill the value inside.

There are two kinds of files, you can download, are listed as follow:

- QSL Templates (System Template and Service Template in XML);
- QSL Documents (Specification specified by QSL in XML);
- QSL Schemas (XSD);

A recommendation of the tool for supporting to edit QSL templates, we used XML Spy to design it, and edit it based on templates to create specifications. In future, we are planning to implement a QSL structure editor to edit specification in a QSL-format, and a QSL generator system to automatically generating e-questionnaire, e-testing, and e-voting systems.

A.2 QSL Grammar Outline

A.2.1 Structure

The Questionnaire Specification Language specification defines a vocabulary and divided messages. Thus common requirements of e-questionnaire, e-testing, and e-voting are defined as elements in the 300-series schemas. The 270 schemas also contain data definition so that be used in data portability. Each message is specified and defined with a separate schema document and can be combined.

As we introduced in pre-chapter, each schema is defined as messages with its unique Ids. According to the combination of messages, users can get the data what they want. Obviously, It is order by a rule we gave to.

A.2.2 Viewing Schemas

QSL grammar is supplied as XML documents. For viewing the structure of it, we recommend using an XML editor, such as XML Spy. Alternatively, if you read a text document with a graphical display, it is easy to refer to.

A.2.3 Schema Diagrams

The diagram A.92 below represents a simple schema in a XML Spy 2011 style. The root element of an instance described by this schema is the element A. The content model of this element is a sequence of the elements B, D and E. The element B is of complex data type *Bstructure*. This contains a choice of either element C or element F. Element C is a restriction of another complex data type *Cstructure*. In this case, the restriction is to forbid the use of the element G (which is defined in *Cstructure* as optional). The other elements allowed are H, which can appear any number of times (but must appear at least once), and I, which can appear up to three times (or not at all). Element D is optional, and of data type *Dstructure*. This has a content model requiring all of the elements J and K, which are both of type *xs:string*. Finally, element E is of simple data type *Etype*, which is restricted from the *xs:NMTOKEN* data type by only allowing the values "yes" and "no".

It is important to remember that these diagrams do not include any attributes. In this document, these are shown in tables below the diagrams.

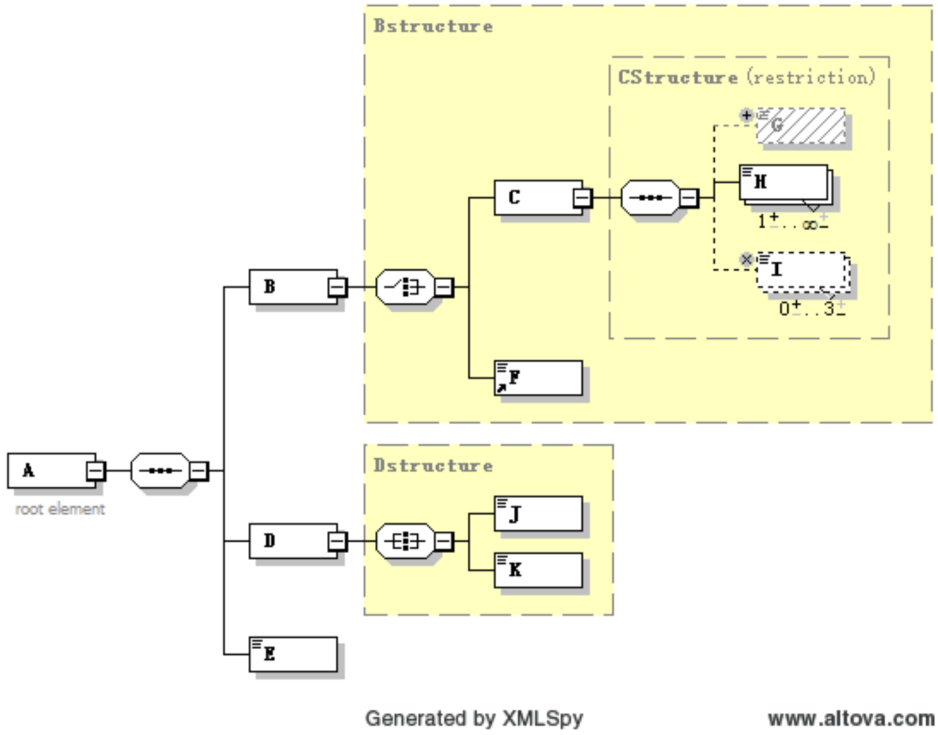

Figure A.1: Sample schema diagram.

A.2.4 Namespaces

The QSL schemas are associated with the namespace: http://www.aise.ics.saitama-u.ac.jp/qsl. This is defined using the prefix qsl. The XML schema namespace http://www.w3.org/2001/XMLSchema is identified by the prefix xs and the XML schema.

Instance namespace: http://www.w3.org/2001/XMLSchema-instance by the prefix xsi.

Schema version is 3.1. This specification is an English version. Such messages start with a QSL element, such as:

```
<?xml version=''1.0" encoding="UTF-8"?>
<QSL
schemaVersion=''3.1"
xml:lang=''en"
xmlns=''http://www.aise.ics.saitama-u.ac.jp/qsl"
xmlns:xs=''http://www.w3.org/2001/XMLSchema"
xmlns:xsi=''http://www.w3.org/2001/XMLSchema-instance">
</QSL>
```

A.2.5 Conventions

Within this specification, the following conventions are used throughout:

- Diagrams are shown as generated by XML Spy 2011 which was also used to generate the schemas and samples. These diagrams show element content, but not attributes;
- Elements and attributes in schemas are identified by partial XPath expressions. Enough of a path is used to identify the item without putting in a full path.

A.3 QSL Core Component

A.3.1 Terminology of QSL

The following terminology is defined in the QSL schema.

1. **Accuracy**: The state of being exact or correct for result
2. **Action**: Destination of logic
3. **Alignment**: Arrangement for sections, questions, and answers
4. **Analyst**: Participant role whose job involves analyze the responses
5. **Analyze**: Function to analyze the responses
6. **Analyzing**: Phase to analyze the responses
7. **Anonymity**: Security about the state of remaining unknown to most other people
8. **Answer**: Function to answer the questions
9. **Authenticate**: Function to prove whether participants are legal or not

10. **Authentication**: Security about proving whether participants are legal or not
11. **Authority**: Security about permission to do something
12. **AutoSaving**: Setting about automatic saving responses
13. **Audit**: Function to see the results are true and correct
14. **Auditing**: Phase to see the results are true and correct
15. **Auditor**: Participant role to see the results are true and correct
16. **Biometric**: Security about Authenticating biometrics
17. **Candidate**: Participant role proposed and considered suitable for an election
18. **Channel**: Security about encryption and communication channel
19. **Checking**: Setting about checking the mistakes
20. **Condition**: A logic rule or decision
21. **Count**: Function to count the numbers of responses and get results
22. **Counting**: Phase to count the numbers of responses and get results
23. **Collect**: Function to collect the responses and store in database
24. **Collecting**: Phase to collect the responses and store in database
25. **Data**: Group for information that should be recorded in the database
26. **Description**: Description title and summary for paper, section, question, and answer
27. **Device**: Environment about devices
28. **Distribute**: Function to distribute paper to legal respondents
29. **Distributing**: Phase to distribute paper to legal respondents
30. **Export**: Function to export data such as paper, setting, response, result
31. **Environment**: Group for a set of environment requirement
32. **Formula**: Numbers or symbols that represent a rule in fields of math, chemical, etc.
33. **Field**: Data schema about part of a record that is a separate item of data
34. **Function**: Group for function requirements

35. **Question**: An unit of section
36. **Questioner**: Participant role who designs a paper sheet and settings
37. **Quota**: Setting limitation about the limited number or amount of respondent
38. **Ratio**: The percentage of being exact or correct for result
39. **Gateway**: Environment for network
40. **Generate**: Function to generate the registration tickets
41. **Import**: Function to import data such as paper, setting, response, result
42. **Integrate**: Function to integrate data
43. **Interval**: Setting about interval answer
44. **Language**: Setting about multi-language
45. **Launch**: Function to launch paper as active state
46. **Limitation**: Limitation for text length and type
47. **Logic**: A set of principles used in setting the relationships of questions
48. **Mark**: Function to give marks to students' work
49. **Marker**: Participant role who mark the exam papers, usually a teacher
50. **Marking**: Phase to give marks to students' work
51. **Media**: Muti-media inserted in the paper
52. **Method**: Analysis methods according to question types
53. **Monitor**: Participant role who monitor the submitting phase
54. **Number**: Setting about ordering the papers, sections, questions, and answers
55. **Observe**: Function to observe the states of respondents
56. **Paper**: Group for a set of questions
57. **Participant**: Group for the roles who take part during the phase
58. **Phase**: Group for process, flow, or procedure
59. **Proposer**: Participant role who formally suggests candidate at an election
60. **Ping**: Function to ping IP address

61. **QSL**: QSL document declaration
62. **Register**: Function to register to the service
63. **Registering**: Phase to register to the service
64. **Remind**: Function to remind and notify participants to do something
65. **Reply**: Data information for each respondent's response
66. **Report**: Function to report the results
67. **Respondent**: Participant role who is the target to be interviewed
68. **Response**: Data information for responses
69. **Result**: Data information for results
70. **Route**: A subset of logic condition
71. **Sample**: A small amount or example of correct answer
72. **Save**: Function to save the paper, setting, answers
73. **Score**: The number of points, goals, etc.
74. **Seal**: Security about encryption
75. **Secrecy**: Security about authentication of something you know such as password
76. **Section**: An unit of paper
77. **Security**: Group for a set of security requirements
78. **Setting**: Group for a set of setting requirements
79. **Server**: Environment about server information
80. **Service**: Group for a set of service requirements
81. **SettingUp**: Phase to set up the research containing paper and settings
82. **Software**: Environment about software information
83. **Solution**: Environment about client-side software solutions
84. **Sponsor**: Participant role who organizes event about research
85. **Statistic**: Data information about the result statistic
86. **Statistics**: The parent element of Statistic
87. **Stop**: Function to stop an activity and the paper state is "closed"

88. **Submit**: Function to submit the responses to store in database
89. **Submitting**: Phase to submit the responses to store in database
90. **System**: Group for a set of system requirements
91. **Time**: Setting about time zone, start time, and end time
92. **Token**: Security about authentication of something you have such as ID card
93. **Upload**: Function to upload files

A.3.2 Simple Types

1. **AddressType**:

   Restriction: xs:token.

   Content: xs:maxLength: 255 with restriction.

   This type is a simple definition of an address.

2. **AuthType**:

   Restriction: xs:token.

   Content: xs: enumeration with restriction.

   This type is a simple definition of authentication methods, which can be classified into 3 types.

Type	Values
Secrecy	password, randomized password
Token	ID card, job card, library card, roll card
Biometric	finger prints, iris, face recognition, signature, DNA
Others	others

3. **BrowserType**:

   Restriction: xs:token.

   Content: xs: enumeration: IE, firefox, sarfi, chrome, others.

   This type is a simple definition of browser. Values are IE, firefox, sarfi, chrome, others.

4. **ChannelType**:

   Restriction: xs:token.

   Content: xs: enumeration: SMS, WAP, digital TV, Internet, Intranet, Kiosk, postal, telephone, digital storage device, paper, fax, email, website, abroad postal, abroad electronic, abroad other, N/A, other.

This type exists to hold the possible enumerations for the channel through where a paper and a register-sheet are submitted, a response is collected. SMS is the short message service (text message). WAP is the wireless access protocol. If other is used, it is assumed that those managing the event will have a common understanding of the channel in use.

5. **ClientSideSolutionType**:

   Restriction: xs:token.

   Content: xs: enumeration: web browser, fat-client, thin-client.

   This type is a simple definition of a browser. Web browser approach is only used to establish the link to run on the submitting server. The fat-client approach is rich in security functionality and a cryptographic algorithm that the client side software needs to be installed and executed on the submitting devices. The thin-client approach is a mix of web browser and fat-client that implements a java applet running in he web browser.

6. **DatabaseType**:

   Restriction: xs:token.

   Content: xs: enumeration: participant, response, report, setting, paper, paper and setting, field, others.

   This simple type is used to define the data type for combinations.

7. **DescriptionType**:

   Restriction: xs:token.

   Content: xs: enumeration: header, paragraph, break.

   This type is a simple definition of description in paper design. This allows that paper, section, question, and answer have herder as a title, paragraph as an explanation, and a break for dividing pages. Considering the lightweight arrangement, which questioner wants, it is designed like this.

8. **DesktopAppType**:

   Restriction: xs:token.

   This type is a simple definition for desktop application.

9. **DeviceType**:

   Restriction: xs:token.

   Content: xs: enumeration: PC, PDA, smart phone, tablet, fable, access point, hub, cable, usb, others.

   This type is a simple definition for the client-side device.

10. **EmailType**:

Restriction: xs:token.

Content: xs: maxLength: 129, and xs:pattern: [^@]+@[^@]+

This type is a simple definition of an email address, pending a more complete description that is widely accepted in government. It allows any characters except the @ symbol, following by an @ symbol and another set of characters.

11. **FieldType**:

    Restriction: xs:token.

    Content: xs: enumeration: char, varchar, number, date, int, float, double, decimal, longtext.

    This type is a simple definition of the field type of database. It is an optional simple type.

12. **FormatType**:

    Restriction: xs:token.

    Content: xs: enumeration: csv, excel, qsl, xml, word, spss, zip, others.

    This type is a simple definition of the format type for export and import.

13. **GenderType**:

    Restriction: xs:token.

    Content: xs: enumeration: male, female, unknown.

    This type is a simple definition of the gender of participant. Options are male, female, or unknown (is not allowed in all contexts).

14. **IdType**:

    Restriction: xs:token.

    Content: Numbers are limited as xs: minLength: 3 and xs: maxLength: 10, and xs: pattern: prefix +[0-9]*.

    This type is a simple definition of QSL elements' Id. To distinguish different elements, it allows the pattern value to add prefixes. The representative prefixes concern the PaperType and ParticipantType.

Element	Pattern	Element	Pattern
Paper	paper[0-9]∗	Section	section[0-9]∗
Answer	answer[0-9]∗	Description	de[0-9]∗
Sponsor	sp[0-9]∗	Respondent	re[0-9]∗
Analyst	an[0-9]∗	Marker	ma[0-9]∗
Auditor	au[0-9]∗	Function	func[0-9]∗
Condition	co[0-9]∗	Action	ac[0-9]∗
Server	server[0-9]∗	Device	device[0-9]∗
Interface	inter[0-9]∗	Statistic	st[0-9]∗
Question	q[0-9]∗	Media	me[0-9]∗
Questioner	qu[0-9]∗	Monitor	mo[0-9]∗
Route	route[0-9]∗	Software	soft[0-9]∗
Method	me[0-9]∗		

15. **LanguageType**:

    Restriction: xs:language.

    This type is a simple definition of declaration of the type of multi-languages.

16. **LimitationType**:

    Restriction: xs:token.

    Content: xs: enumeration: alphabet, figure.

    This type is a simple definition of limitation type.

17. **LogicType**:

    Restriction: xs:token.

    Content: xs: enumeration: skipping, piping, extraction, randomization.

    This type is a simple definition of logic type. Its values are basic logic type. As to the combination of them are introduce in Logic Chapter in details. This manual takes a complete chapter to explain it. It is a difficult part in this manual.

18. **MediaType**:

    Restriction: xs:token.

    Content: xs: enumeration: video/ogg, video/mp4, audio/ogg, audio/mpeg, img/jpg, img/png, img/gif.

    This type is a simple definition of media types. It is designed by referring to HTML's tags and "video" and "img."

19. **NameType**:

    Restriction: xs:token.

    Content: xs: maxLength: 129, and xs: maxLength: 2.

This type is a simple definition of the name of participant and affiliation.

20. **OrderType**:

    Restriction: xs:token.

    Content: xs: enumeration: ascending, descending, random.

    This type is a simple definition of the order type of settings.

21. **OSType**:

    Restriction: xs:token.

    Content: xs: enumeration: Windows XP, Win7, Windows 2003, Windows Vista, Win8, Win10, ios 10.0, ubuntu 14.04, Andirod, others.

    This type is a simple definition of operating systems (usually defined in software).

22. **PaperType**:

    Restriction: xs:token.

    Content: xs: enumeration: sponsor, questioner, respondent, marker, auditor, analyst, monitor, candidate, proposer.

    This type is a simple definition of the participant roles.

23. **PhaseType**:

    Restriction: xs:token.

    Content: xs: enumeration: setting up, submitting, registering, collecting, analyzing, marking, auditing, counting.

    This type is a simple definition of phase.

24. **QuestionType**:

    Restriction: xs:token.

    This type is a simple definition of question types. It is much more complex because until now there are over 36 question types in the world. Much more detailed information is defined in Chapter about Question Types.

25. **ReminderType**:

    Restriction: xs:token.

    Content: xs: enumeration: password, token type, answer notification, answer finished, close submitting time, declaration time, report.

    This type is a simple definition to notify what contents.

26. **ReportType**:

    Restriction: xs:token.

Content: xs: enumeration: basic, gap, trend.

This type is a simple definition of analysis types. Basic type is a percentage report. Gap type is the comparison report. Trend type is always used in e-questionnaire for trend report.

27. **SoftwareType**:

    Restriction: xs:token.

    Content: xs: enumeration: database, browser, desktop app, OS.

28. **SystemType**:

    Restriction: xs:token.

    Content: xs: enumeration: system.

    If it needs to be divided into 3e systems for extensions.

29. **TelNumType**:

    Restriction: xs:token.

    Content: xs: maxLength: 35, xs: minLength: 1, and xs: pattern: [0-9]1,35.

    Since this must allow for various styles of international telephone number. The pattern has been kept simple. This allows an option plus sign, then between 1 and 35 characters with a combination of digits, brackets, the dash symbol and white space.

30. **TimeType**:

    Restriction: xs:dateTime.

    This type abides by the format of XML Schema's xs:dateTime. The definition of the date and time uses the following format: YYYY-MM-DDThh:mm:ss Z(YYYY stands for year, MM stands for month, DD stands for day, T stands for the start of necessary time, hh stands for hour, mm stands for minute, ss stands for second, Z stands for the time zone identified by UTC).

31. **ReportType**:

    Restriction: xs:token.

    Content: xs:enumeration: yes, no.

    This is a simple enumeration of yes and no and is used for elements and attributes than can only take these values.

A.4 QSL Schema Descriptions

A.4.1 Overview

The following items are listed as an overview of QSL schemas. Further explanations are given in the following subsections.

1. **QSL-100-QSL** declares QSL documents
2. **QSL-110-System** provides schema of system
3. **QSL-120-Service** provides schema of service
4. **QSL-210-Phase** provides schema for complete phases with fulfilled functions and settings
5. **QSL-220-Security** provides schema for system security requirements
6. **QSL-230-Paper** provides schema of a complete paper construction with logic
7. **QSL-240-Setting** provides schema of settings that questioner sets up before distributing
8. **QSL-250-Environment** provides schema of server, software, device, network, interface
9. **QSL-260-Participant** provides schema of participant roles and information
10. **QSL-270-Function** provides schema of system functions
11. **QSL-280-Data** provides schema of recorded data of services, and database fields for systems
12. **QSL-310-SettingUp** provides schema of setting up phase
13. **QSL-311-Registering** provides schema of registering phase
14. **QSL-312-Distributing** provides schema of distributing phase
15. **QSL-313-Submitting** provides schema of submitting phase
16. **QSL-314-Collecting** provides schema of collecting phase
17. **QSL-315-Analyzing** provides schema of analyzing phase
18. **QSL-316-Counting** provides schema of counting phase
19. **QSL-320-Sponsor** provides schema of participant role of sponsor
20. **QSL-321-Questioner** provides schema of participant role of questioner
21. **QSL-322-Respondent** provides schema of participant role of respondent

22. **QSL-323-Analyst** provides schema of participant role of analyst
23. **QSL-324-Monitor** provides schema of participant role of monitor
24. **QSL-330-Export** provides schema of export function
25. **QSL-331-Import** provides schema of import function
26. **QSL-332-Launch** provides schema of launch function
27. **QSL-333-Stop** provides schema of stop launching function
28. **QSL-334-Generate** provides schema of generate function
29. **QSL-335-Ping** provides schema of ping IP function
30. **QSL-336-Integrate** provides schema of integrate function
31. **QSL-337-Remind** provides schema of remind function
32. **QSL-340-Server** provides schema of server information
33. **QSL-341-Field** provides schema of database field for system data
34. **QSL-342-Gateway** provides schema of network information
35. **QSL-343-Interface** provides schema of interface information
36. **QSL-344-Device** provides schema of device information
37. **QSL-345-Software** provides schema of software information
38. **QSL-350-Anonymity** provides schema of anonymous methods
39. **QSL-351-Authentication** provides schema of authentication with methods and channel
40. **QSL-352-Authority** provides schema of access control
41. **QSL-353-Seal** provides schema of encryption
42. **QSL-354-Channel** provides schema of communication channel
43. **QSL-360-Section** provides schema of section for a group of questions
44. **QSL-361-Question** provides schema of question for a group of answers
45. **QSL-362-Answer** provides schema of answer
46. **QSL-363-Description** provides schema of description for paper, section, question, and answer
47. **QSL-364-Media** provides schema of multi-media

48. **QSL-365-Alignment** provides schema of paper arrangement
49. **QSL-366-Limitation** provides schema of limitation of text or numbers
50. **QSL-370-Language** provides schema of setting for multi-languages
51. **QSL-371-Time** provides schema of start time, end time, and time zone
52. **QSL-372-Number** provides schema of numbering following orders
53. **QSL-373-Quota** provides schema of limitation for rate, quota, or numbers of respondents
54. **QSL-374-Response** provides schema of response data
55. **QSL-375-Result** provides schema of result data
56. **QSL-410-Logic** provides schema of logic of paper, section, question, and answer
57. **QSL-510-Marking** provides schema of marking phase
58. **QSL-520-Score** provides schema of scoring rules
59. **QSL-530-Sample** provides schema of sample answers
60. **QSL-540-Formula** provides schema of math formula
61. **QSL-550-Marker** provides schema of marker and information
62. **QSL-610-Auditing** provides schema of auditing phase
63. **QSL-620-Candidate** provides schema of candidate and information
64. **QSL-630-Proposer** provides schema of proposer information
65. **QSL-640-Auditor** provides schema of auditor information
66. **QSL-710-simpleType** provides general simpleType definitions for reusing attribute values

A.4.2 100-QSL

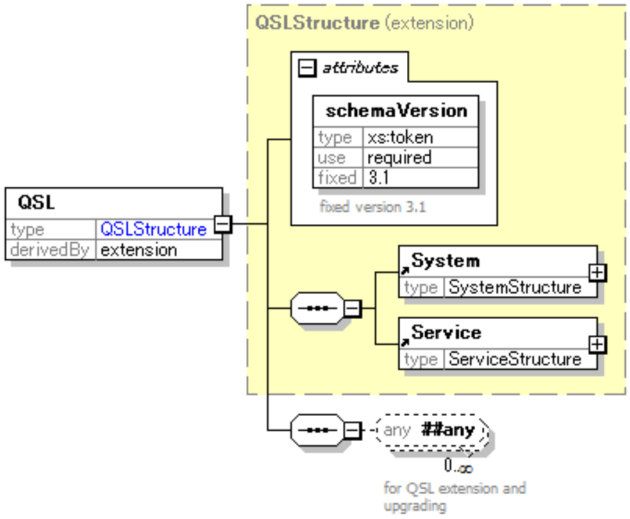

Figure A.2: QSL diagram.

Fig. A.2 shows the schema of **QSL**. This schema is used for declaration of a specification, as well as the identifier for QSL structure editor. Defined any specification must define this schema. This schema is based on the extension of complex type **QSLStructure**.

Element	Attribute	Type	Use	Fixed
QSL	schemaVersion	xs:token	required	3.1

A.4.3 110-System

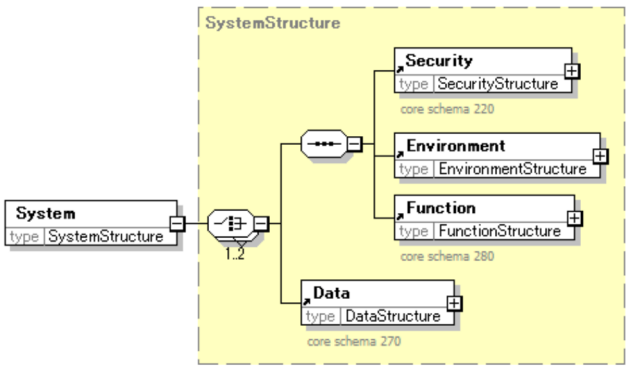

Figure A.3: System diagram.

Fig. A.3 shows the schema of **System**. This schema is used for specifying the system, which provides services to do e-questionnaire, e-testing, and e-voting.

It specifies the functions during each phase, and component information such as server, interface, database, device, etc. This schema is based on the complex type **SystemStructure**.

A.4.4 120-Service

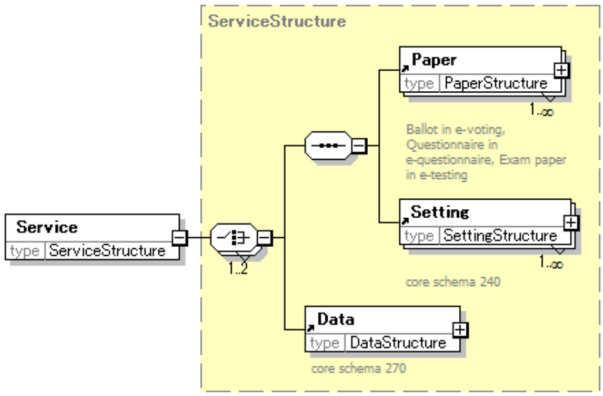

Figure A.4: Service diagram.

Fig. A.4 shows the schema of **Service**. This schema records the service consisting of lots of paper sheets, and settings for papers. Questioners and sponsors can directly use its specifications to define papers to respondents. It is the core component of the question bank for e-questionnaire and e-testing. This schema is based on complex type of **ServiceStructure**.

A.4.5 210-Phase

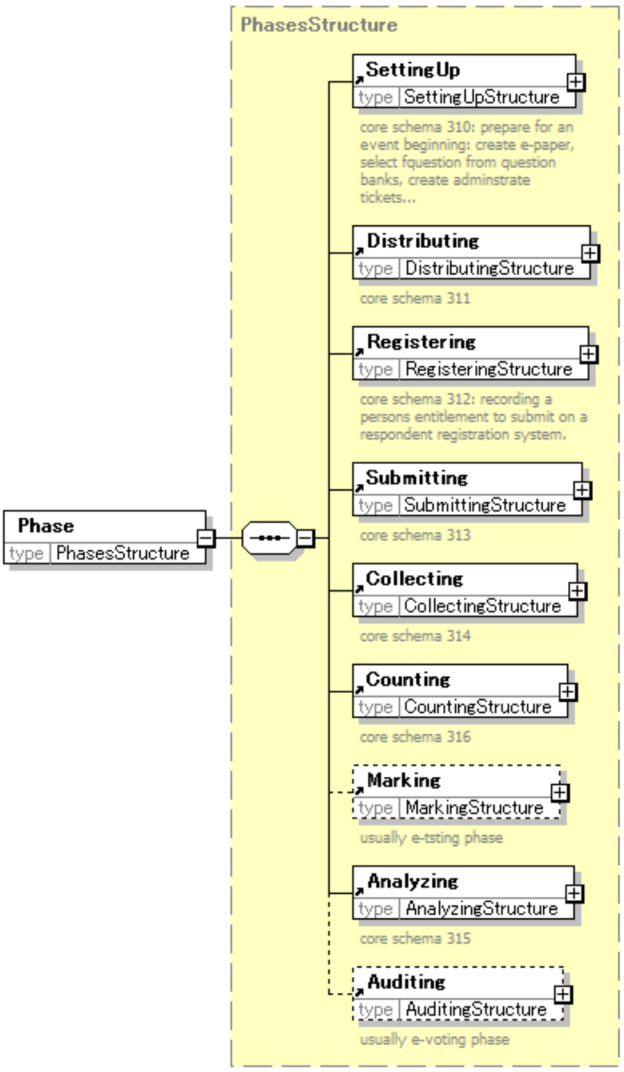

Figure A.5: Phase diagram.

Fig. A.5 shows the schema of **Phase**. This schema is used to specify phases. All the child elements should be described by a sequence order. And the child elements are based on schema 310, 311, 312, 313, 314, 315, and 316. This schema is based on complex type of **PhaseStructure**.

For e-questionnaire, it is by the order of 310, 311, 312, 313, 314, 315, and 316.

For e-testing, it is by the order of 310, 311, 312, 313, 314, 315, 510, and 316.

For e-voting, it is by the order of 310, 311, 312, 313, 314, 315, 316 and 610.

A.4.6 220-Security

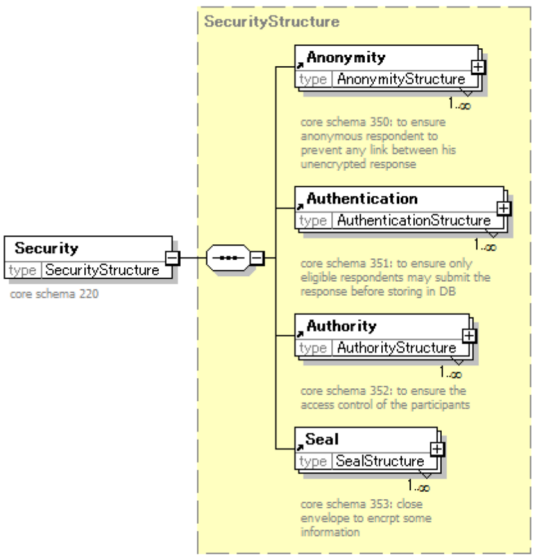

Figure A.6: Security diagram.

Fig. A.6 shows the schema of **Security**. This schema is used to specify system security consisting of schema 350, 351, 352, and 353. This schema is based on the complex type **SecurityStructure**.

A.4.7 230-Paper

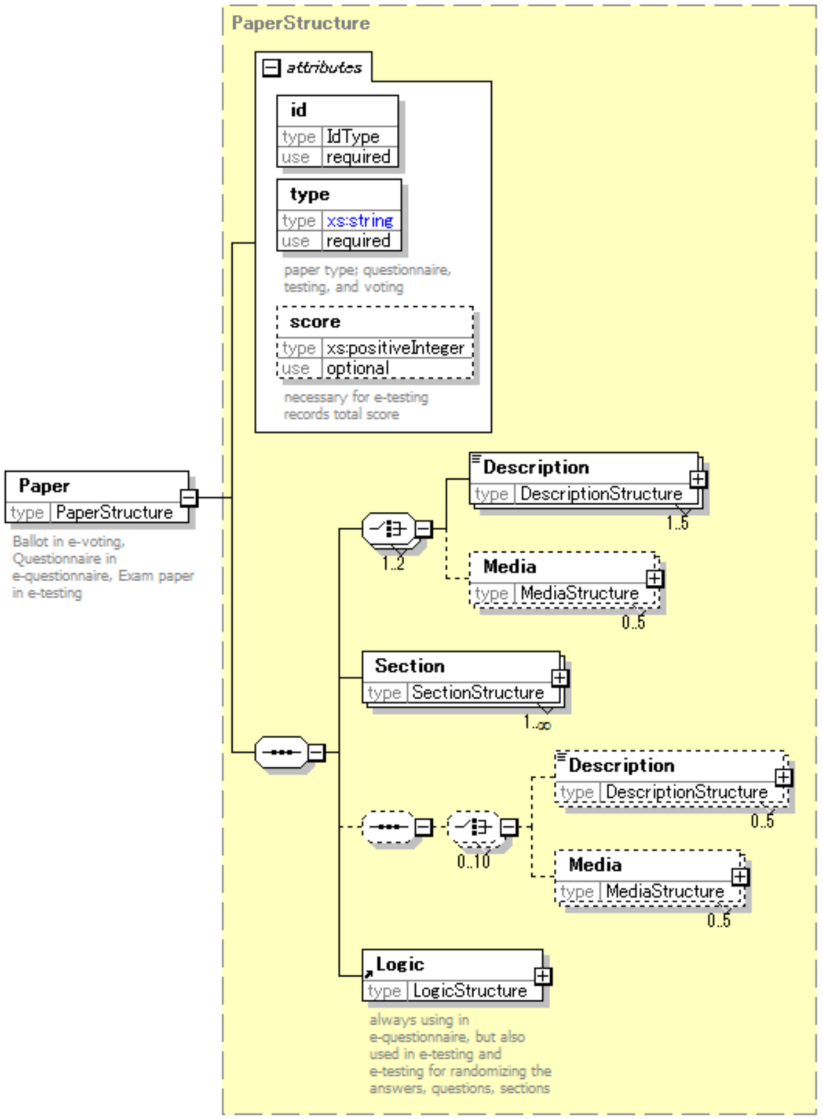

Figure A.7: Paper diagram.

Fig. A.7 shows the schema of **Paper**. This schema is used to specify paper sheet. There are 3 attributes. This schema is based on the complex type **PaperStructure**.

Element	Attribute	Type	Use	Remark
Paper	id	IdType	required	paper identifier
	type	xs:string	required	classification for e-q,t,v
	score	xs:positiveInteger	optional	total score for e-testing

A.4.8 240-Setting

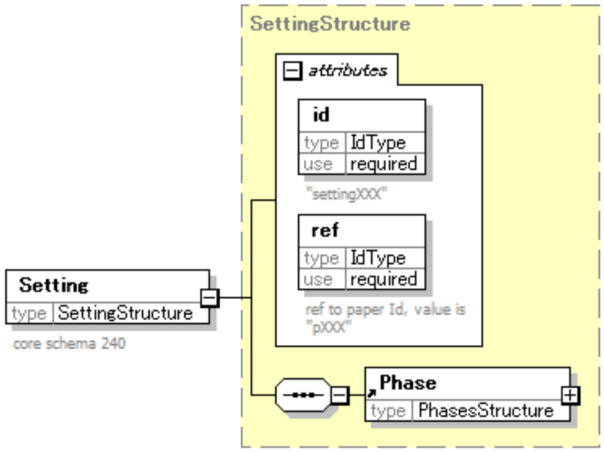

Figure A.8: Setting diagram.

Fig. A.8 shows the schema of **Setting**. This schema is used to specify setting for corresponding paper sheet. It consists of schema 210. This schema is based on the complex type **SettingStructure**.

Element	Attribute	Type	Use	Remark
Setting	id	IdType	required	identifier
	ref	IdType	required	refer to paper

A.4.9 250-Environment

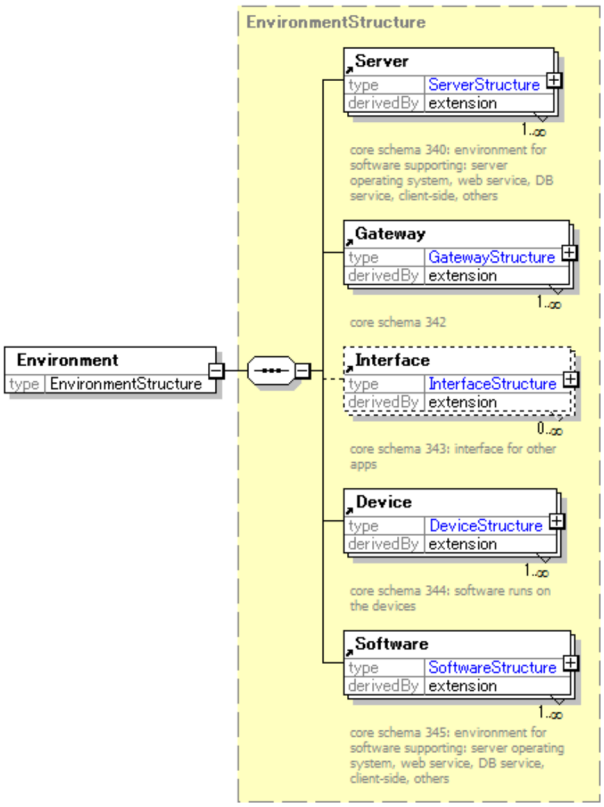

Figure A.9: Environment diagram.

Fig. A.9 shows the schema of **Environment**. This schema is used to specify system environment consisting of schema 340, 342, 343, 344, and 345. This schema is based on the complex type **EnvironmentStructure**.

A.4.10 260-Participant

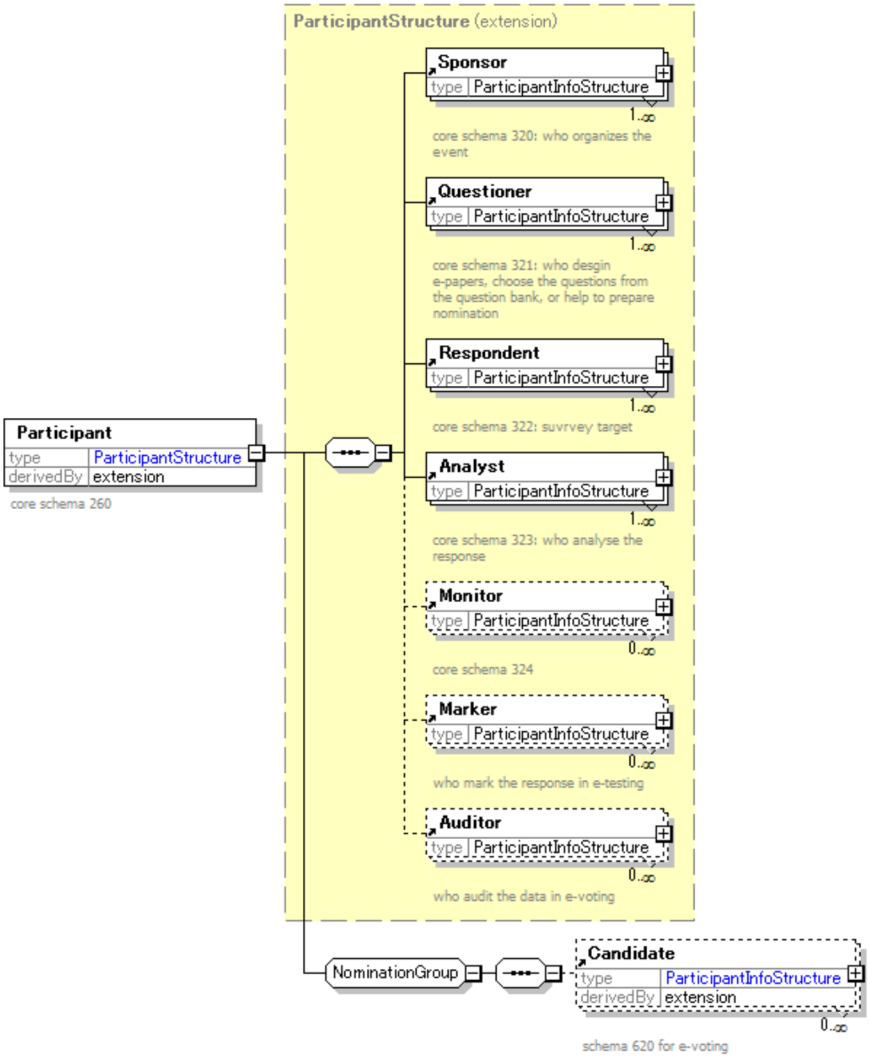

Figure A.10: Participant diagram.

Fig. A.10 shows the schema of **Participant**. This schema is used to specify a list of participant roles. This schema is based on the complex type **DataStructure**. But there is a *NominationGroup* for defining nomination participants containing candidates and its proposers. That means, **Proposer** element (minOcc 1 and MaxOcc unbounded) is the child elements of **Candidate** element.

A.4.11 270-Data

Fig. A.11 shows the schema of **Data**. This schema is used to specify data, which is divided into 2 groups for specifying the system data to define the database fields, and for specifying the service data to record the information. The groups are ordered in the choice relations. When the **Data** tag is the child of System tag, you

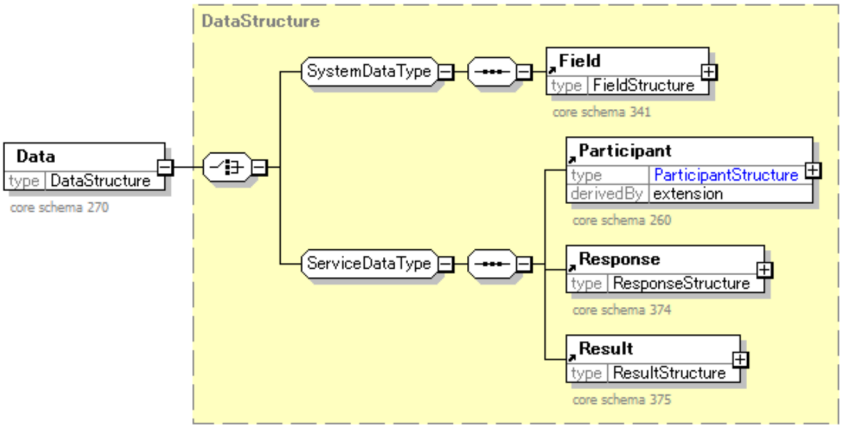

Figure A.11: Data diagram.

can choose the schema for *SystemDataGroup*. Otherwise, you can choose another group. This schema is based on the complex type **DataStructure**.

A.4.12 280-Function

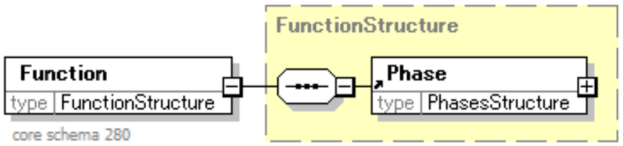

Figure A.12: Function diagram.

Fig. A.12 shows the schema of **Function**. This schema is used to specify system functional requirements consisting of schema 210. This schema is based on the complex type **FunctionStructure**.

A.4.13 310-SettingUp

Fig. A.13 shows the schema of **SettingUp**. This schema is used to specify the phase for preparing an event beginning. This schema is based on the complex type **SettingUpStructure**. Because all the phases are listed in **Function** and **Setting**, We defined each phase has two groups one for function definition, another for setting definition. And the two groups have the choice order relationship. That means, the users choose the set of elements in function group to define function requirements in this phase, and choose the set of elements in setting group to define setting requirements in this phase. The elements in function groups are explained in core schemas (schema 330, 331,332, 333, 336, and 337) and some elements in setting groups are also defined in core schemas (schema 370, 371, 372, and 373), please refer to the corresponding schema definitions. There are 5 elements are not listed in schema files respectively, because they have the duplicated meanings with other tags.

Element	Group	Remark
SettingUp	SettingUpFunctionPhaseGroup	belong to 280 schema
	SettingPhaseGroup	belong to 240 schema

- **AutoSaving**

  Fig. A.14 shows the schema of **AutoSaving**. This schema is used to define to automatically saving the answers when the respondents answering the questions before submitting their responses. The *saveTime* attribute means how many seconds record once. The *closeTime* means when to shut down this function.

Element	Attribute	Type	Use
AutoSaving	enable	YesNoType	required
	scoping	PhaseType	optional
	closeTime	xs:nonNegativeInteger	optional
	saveTime	xs:nonNegativeInteger	optional

- **Checking**

  Fig. A.15 shows the schema of **Checking**. This element defines the spelling checking setting for providing conveniences for respondents to answer questions.

Element	Attribute	Type	Use
Checking	enable	YesNoType	required
	scoping	PaperType	optional

- **Distribute**

  Fig. A.16 shows the schema of **Distribute**. This schema is used to define the distributing method (channel). The users use this element to define that they distribute the paper to respondents in which kinds of the channel. This value must match with the distribute function during distributing phase.

Element	Attribute	Type	Use
Distribute	method	ChannelType	required

- **Analyze**

  Fig. A.32 shows the schema of **Analyze**. **Basic analysis**: provides to analyze by machine automatically. **Gap analysis** (Removed): provides to analyze for side-by-side matrix question type. **Trend analysis**: provides a look at data over time for a long-running paper.

Element	Attribute	Type	Use
Analyze	enable	YesNoType	required
	type	ReportType	optional

- **Interval**

  Fig. A.18 shows the schema of **Interval**. This schema is used to define the interrupt during the phases. The users use this element to define that they allow the respondent to stop and back to answer. The *frequency* attribute is used to record that the respondents are allowed to interrupt how many times.

Element	Attribute	Type	Use
Interval	enable	YesNoType	required
	frequency	union: xs:nonNegativeInteger and xs:token (unbounded)	optional

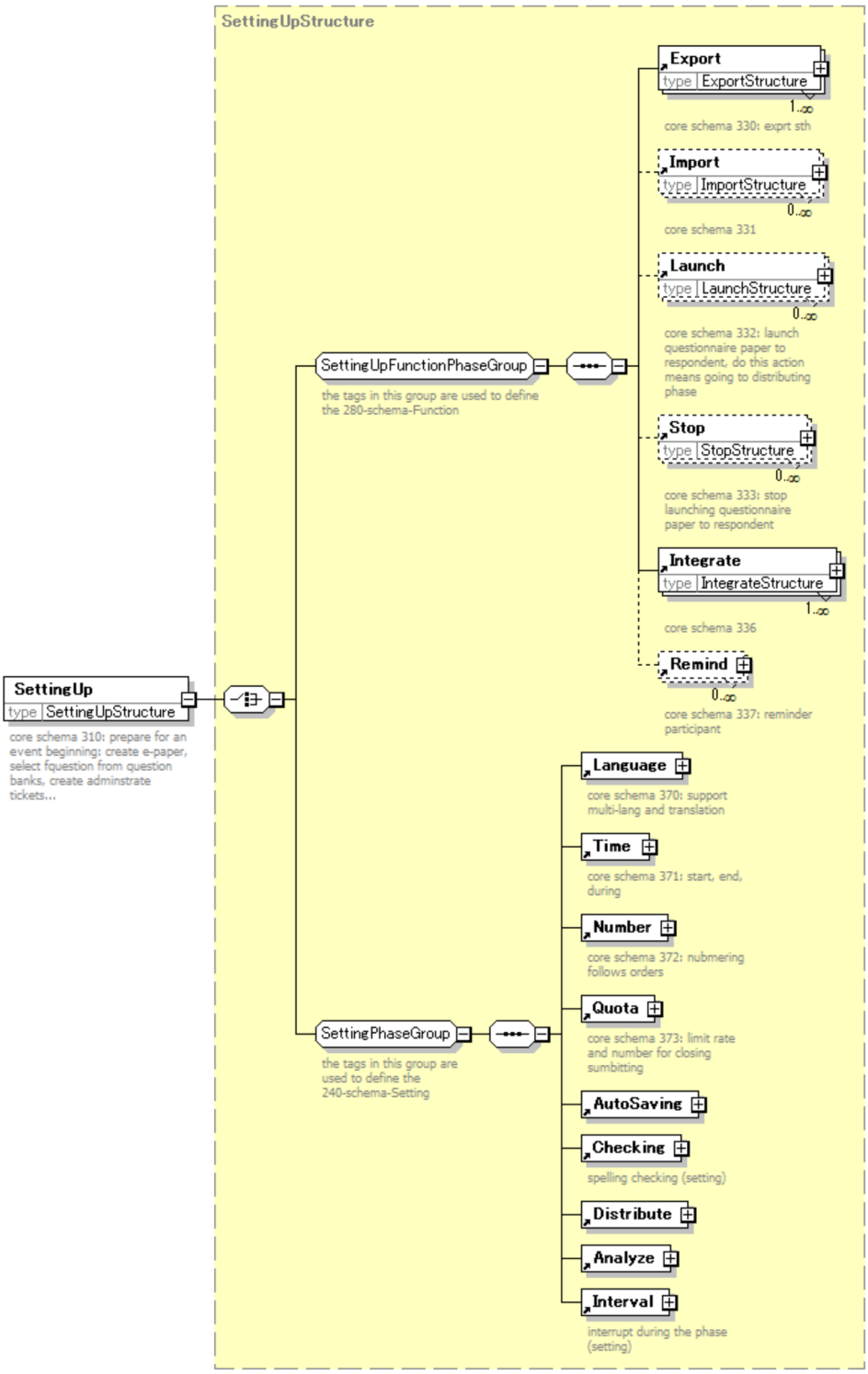

Figure A.13: SettingUp diagram.

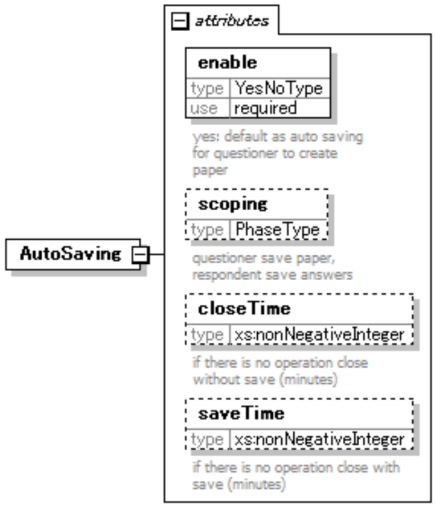

Figure A.14: AutoSaving diagram.

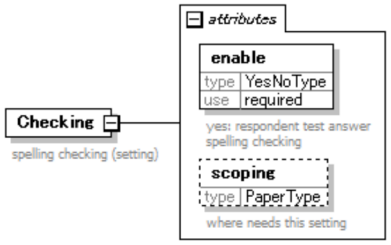

Figure A.15: Checking diagram.

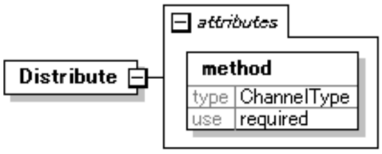

Figure A.16: Distribute diagram.

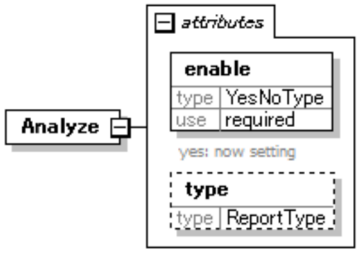

Figure A.17: Analyze diagram.

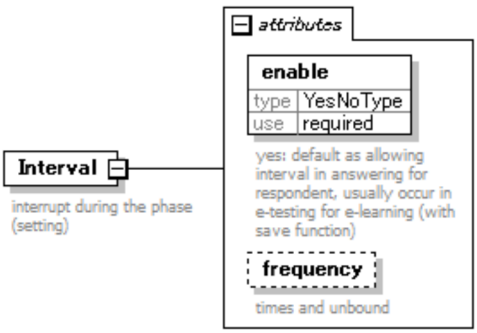

Figure A.18: Interval diagram.

A.4.14 311-Distributing

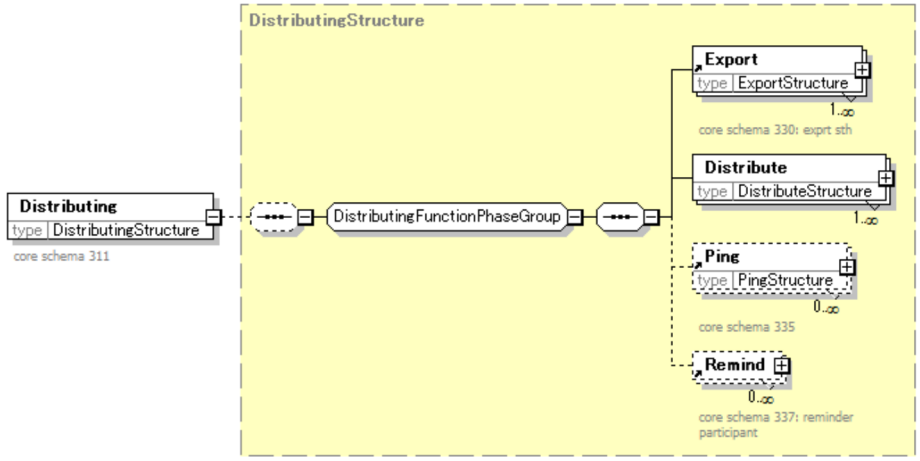

Figure A.19: Distributing diagram.

Fig. A.19 shows the schema of **Distributing**. This schema is used to specify the phase for distributing the paper to respondents or to the web with some access limitation. This schema is based on the complex type **DistributingStructure**. Because all the phases are listed in Function and Setting, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we do not define the settings during this phase. The elements in function groups are explained in core schemas (schema 330, 335, and 337), please refer to the corresponding schema definitions. There is an element that is not listed in the schema file because they have the duplicated meanings with other tags.

Element	Group	Remark
Distributing	DistributingFunctionPhaseGroup	belong to 280 schema

- **Distribute**

  Fig. A.20 shows the schema of **Distribute**. This schema is used to define the distribute function. The users use this element to define that they distribute the paper to respondents by whom in which kinds of the channel. This value must match with the distribute function setting in setting up phase. This schema is based on the complex type of **DistributeStructure**.

Element	Attribute	Type	Use
Distribute	id	IdType	required
	role	ParticipantType	required
	channel	ChannelType	required

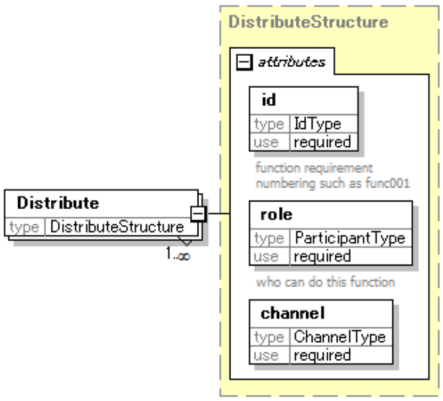

Figure A.20: Distribute diagram.

A.4.15 312-Registering

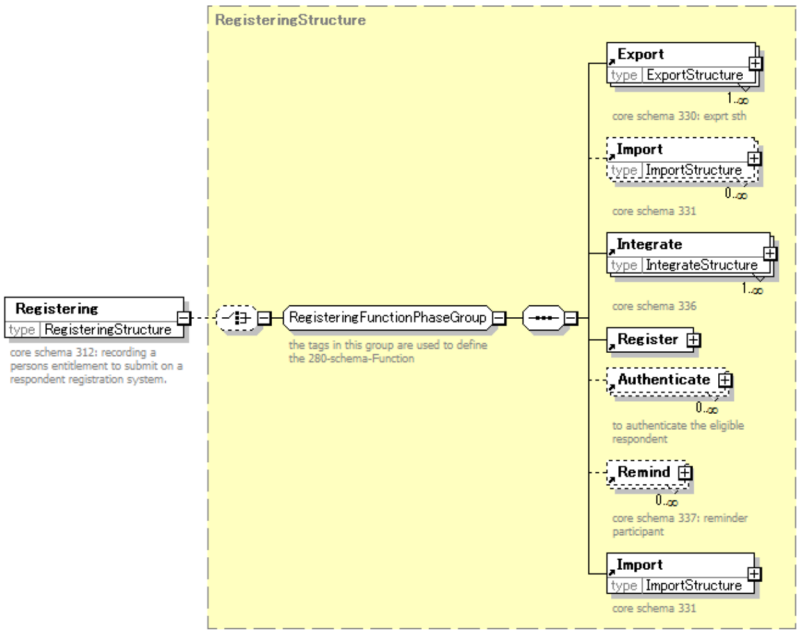

Figure A.21: Registering diagram.

Fig. A.21 shows the schema of **Registering**. This schema is used to specify the phase for respondent registration. This schema is based on the complex type **RegisteringStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we do not define the settings during this phase. The elements in function groups are explained in core schemas (schema 330, 331, 336, and 337), please refer to the corresponding schema definitions. There are 2 elements that are not listed in schema files because they have the duplicated meanings with other tags.

Element	Group	Remark
Registering	RegisteringFunctionPhaseGroup	belong to 280 schema

- **Authenticate**

  Fig. A.22 shows the schema of **Authenticate**. This schema is used to define the authenticate function to verify the eligible respondents. This value must match with the **Authenticate** *method*defined in security requirements.

Element	Attribute	Type	Use
Authenticate	id	IdType	required
	role	ParticipantType	required
	method	AuthType	required

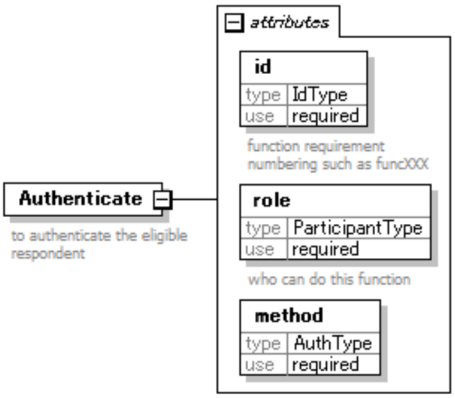

Figure A.22: Authenticate diagram.

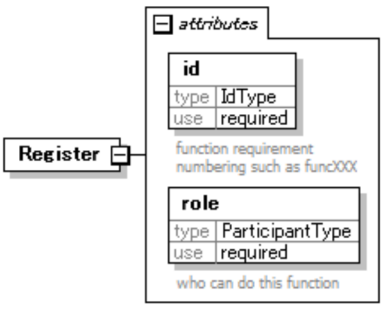

Figure A.23: Register diagram.

- **Register**

  Fig. A.23 shows the schema of **Register**. This schema is used to define the register function. Usually, when the users set everyone can access the survey, so that there is no need to authenticate but need register respondents information to store in database sometimes.

Element	Attribute	Type	Use
Register	id	IdType	required
	role	ParticipantType	required

A.4.16 313-Submitting

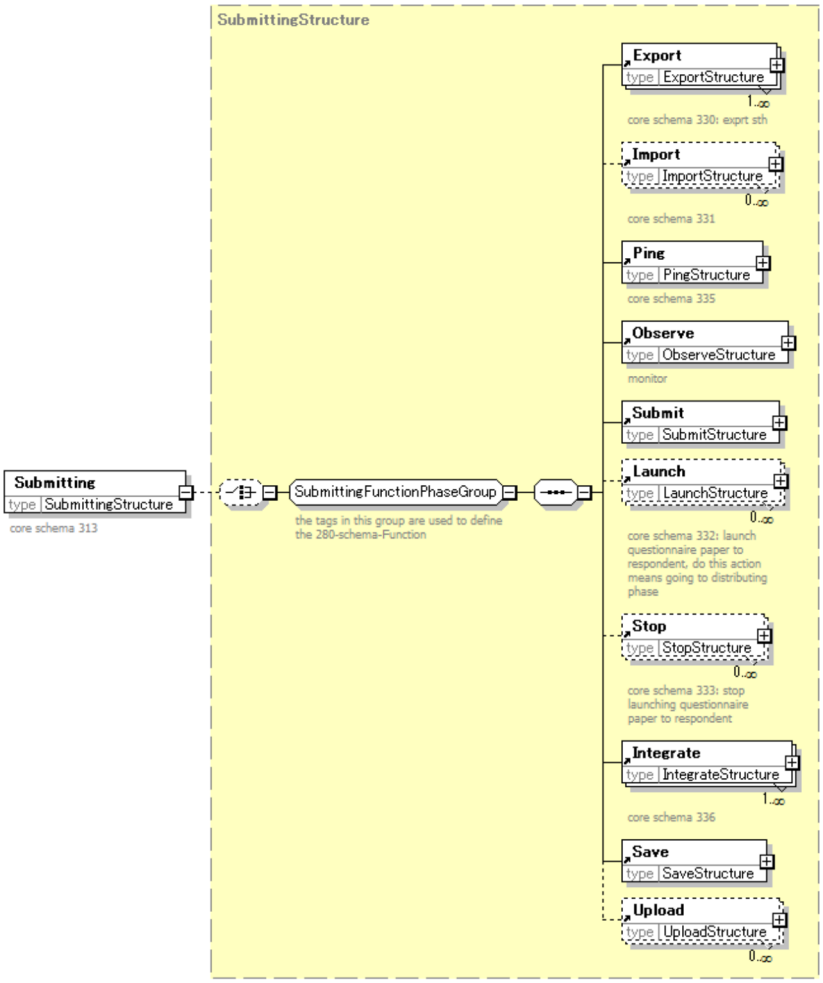

Figure A.24: Submitting diagram.

Fig. A.24 shows the schema of **Submitting**. This schema is used to specify the phase for respondents to submit their responses to store in the database. This schema is based on the complex type **SubmittingStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we do not define the settings during this phase. The elements in function groups are explained in core schemas (schema 330, 331, 332, 333, 335, and 336), please refer to the corresponding schema definitions. There are 4 elements that are not listed in schema files because they have the duplicated meanings with other tags.

Element	Group	Remark
Submitting	SubmittingFunctionPhaseGroup	belong to 280 schema

- **Observe**

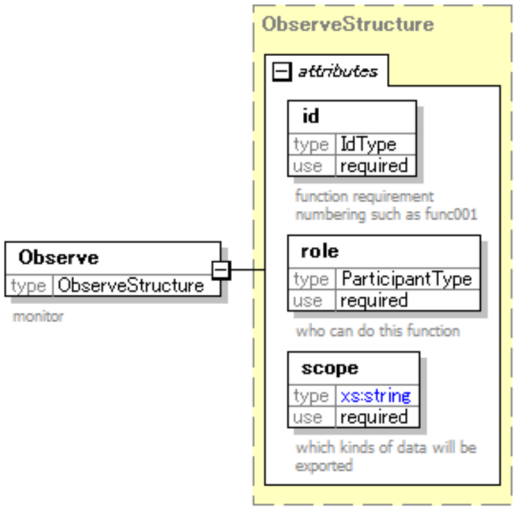

Figure A.25: Observe diagram.

Fig. A.25 shows the schema of **Observe**. This schema is used to define the monitor function. It is usually used for monitors. The monitors can monitor each respondent 's state such as the completeness of answering paper.

Element	Attribute	Type	Use
Observe	id	IdType	required
	role	ParticipantType	required
	scope	derived by xs:string	required

- **Save**

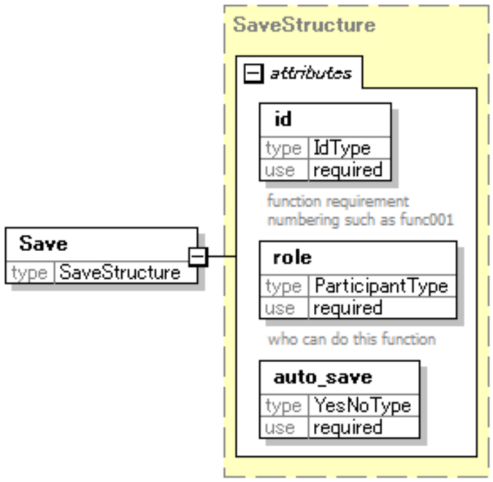

Figure A.26: Save diagram.

Fig. A.26 shows the schema of **Save**. This schema is used to define the save function. Usually, when the respondents are answering the questions, the system can automatically save their responses if the questioners set up **AutoSaving** in **Setting**.

Element	Attribute	Type	Use
Save	id	IdType	required
	role	ParticipantType	required
	autoSave	YesNoType	required

- **Submit**

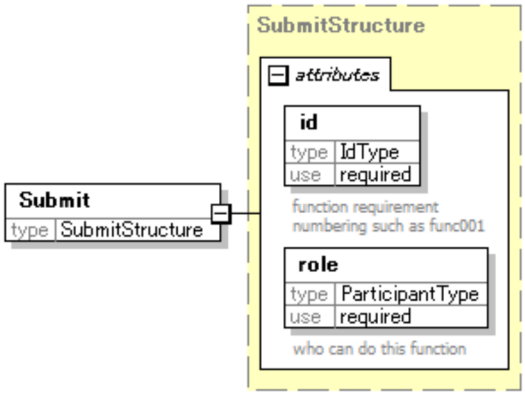

Figure A.27: Submit diagram.

Fig. A.27 shows the schema of **Submit**. This schema is used to define the submit function. The system must provide the interface to submit their responses and save in the database.

Element	Attribute	Type	Use
Save	id	IdType	required
	role	ParticipantType	required

- **Upload**

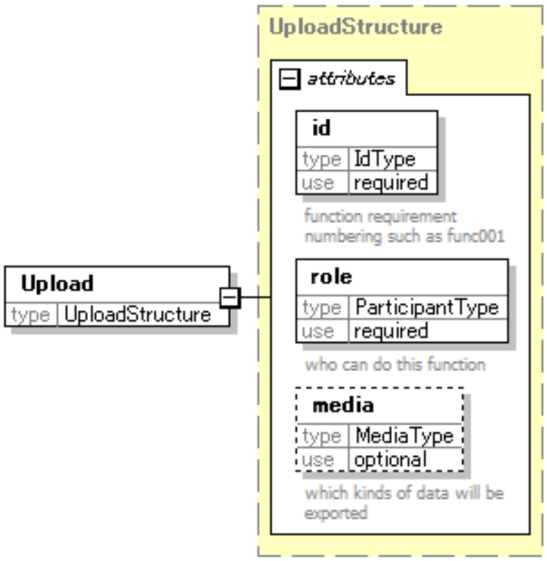

Figure A.28: Upload diagram.

Fig. A.28 shows the schema of **Upload**. This schema is used to define the upload function. Some questions request the respondents to upload their responses.

Element	Attribute	Type	Use
Upload	id	IdType	required
	role	ParticipantType	required
	media	MediaType	optional

A.4.17 314-Collecting

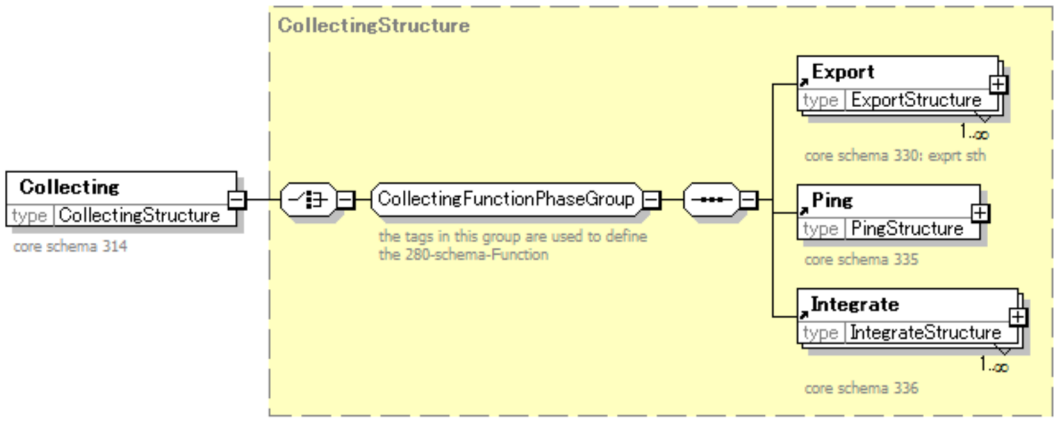

Figure A.29: Collecting diagram.

Fig. A.29 shows the schema about **Collecting**. This schema is used to specify the phase for collecting responses. This schema is based on the complex type **CollectingStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we are not define the settings during this phase. The elements in function groups are explained in core schemas (schema 330, 335, and 336), please refer to the corresponding schema definitions. As the last child element, **Collect** is not listed in schema file, because they have the duplicated meanings with other tags.

Element	Group	Remark
Collecting	CollectingFunctionPhaseGroup	belong to 280 schema

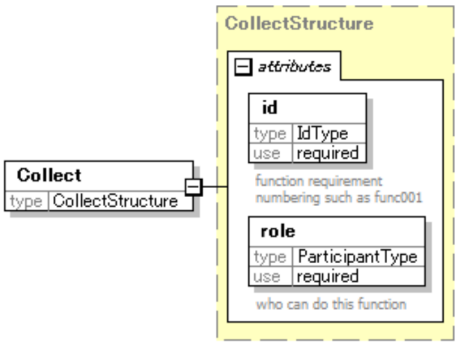

Figure A.30: Collect diagram.

Fig. A.30 shows the schema about **Collect**. This schema is used to define the collect function. The system should provide interface to respondents to collect data to store in the database.

Element	Attribute	Type	Use	Remark
Collect	id	IdType	required	function Id marked as funcXXX
	role	ParticipantType	required	who can do this function

A.4.18 315-Analyzing

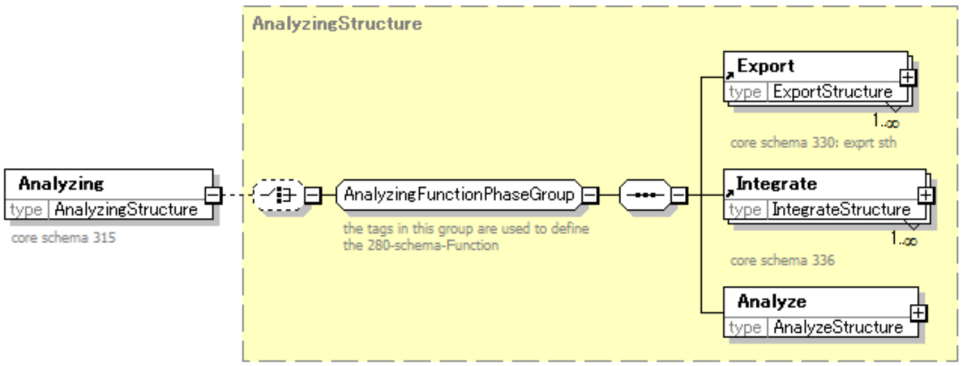

Figure A.31: Analyzing diagram.

Fig. A.31 shows the schema about **Analyzing**. This schema is used to specify the phase for analyzing the collected responses. This schema is based on the complex type **AnalyzingStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we are not define the settings during this phase. The elements in function groups are explained in core schemas (schema 330 and 336), please refer to the corresponding schema definitions. As the last child element, **Analyze** is not listed in schema file, because they have the duplicated meanings with other tags.

Element	Group	Remark
Analyzing	AnalyzingFunctionPhaseGroup	belong to 280 schema

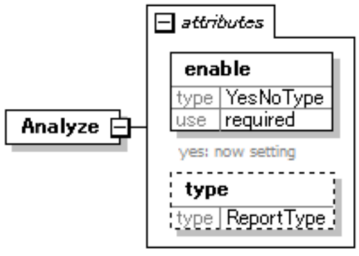

Figure A.32: Analyze diagram.

Fig. A.32 shows the schema about **Analyze**. This schema is used to define the analyze function. The system should provide interface to analyze response data to get a result.

Element	Attribute	Type	Use	Remark
Analyze	id	IdType	required	function Id marked as funcXXX
	role	ParticipantType	required	who can do this function

A.4.19 316-Counting

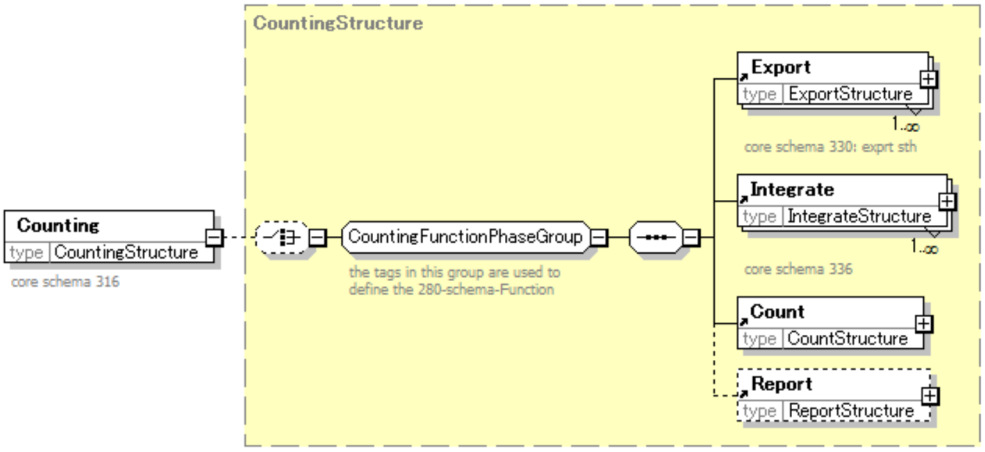

Figure A.33: Counting diagram.

Fig. A.33 shows the schema about **Counting**. This schema is used to specify the phase for counting to get the result. This schema is based on the complex type **CountingStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we are not define the settings during this phase. The elements in function groups are explained in core schemas (schema 330 and 336), please refer to the corresponding schema definitions. As the last child element, **Count** and **Report** are not listed in schema files, because they have the duplicated meanings with other tags.

Element	Group	Remark
Counting	CountingFunctionPhaseGroup	belong to 280 schema

Fig. A.34 shows the schema about **Count**. This schema is used to define the count function. The system should provide interface to count response data to get a result.

Element	Attribute	Type	Use	Remark
Count	id	IdType	required	function Id marked as funcXXX

Fig. A.35 shows the schema about **Report**. This schema is used to define the report function. The system should provide interface to report the result for normal or in trend.

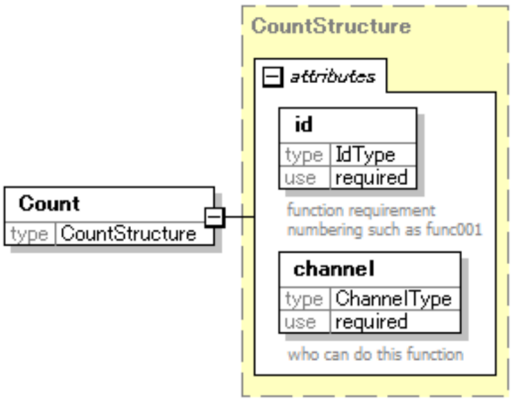

Figure A.34: Count diagram.

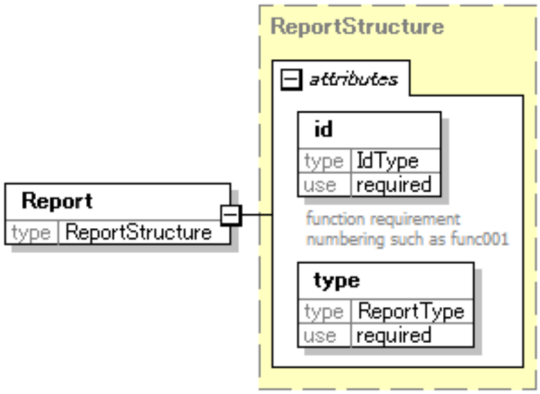

Figure A.35: Report diagram.

Element	Attribute	Type	Use	Remark
Count	id	IdType	required	function Id marked as funcXXX
	type	ReportType	required	basic/ trend

A.4.20 320-Sponsor

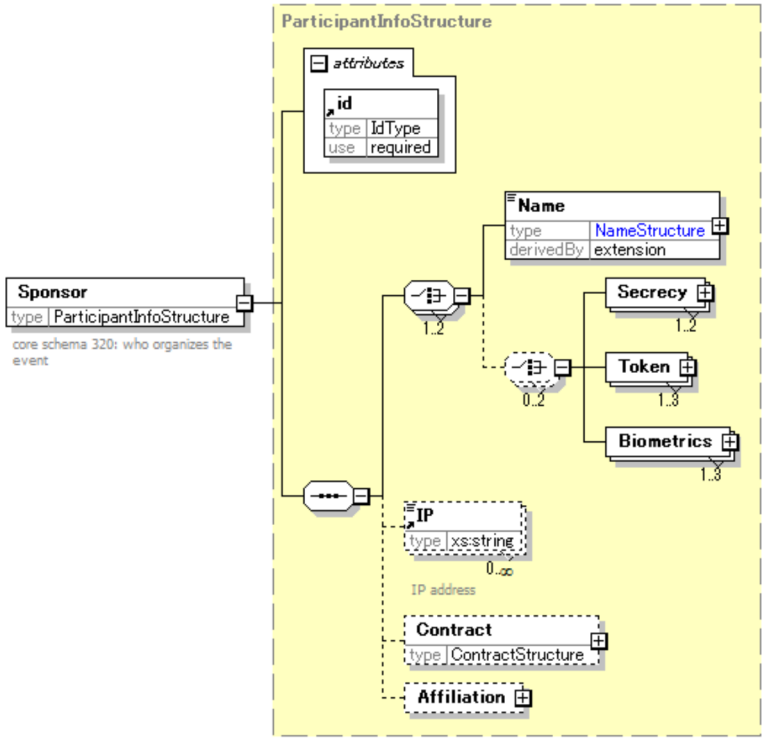

Figure A.36: Sponsor diagram.

Fig. A.36 shows the schema about **Sponsor**. This schema is used to define the sponsor information. The schema is based on the complex type of **ParticipantInfoStructure**. Other roles of participants are also based on this complex type. If the security requirements demands to authenticate, then the role's information should be recorded authentication method and its values. According to the setting about recording IP addresses, the value should be recorded in **IP** element as its content. In addition, some contract information and affiliation information should be recorded, if the developers defined the fields.

Element	Attribute	Type	Use
Sponsor	id	IdType	required
Name	fullName	NameType	required
	usedName	NameType	optional
	firstName	NameType	optional
	familName	NameType	optional
Secrecy	type	AuthType	required
	value	xs:token	required
Token	type	AuthType	required
	value	xs:token	required
Biometric	type	AuthType	required
	value	xs:token	required
IP	-	derived by xs:string	required
Contract	address	AddressType	required
	tel	TelNumType	required
	phoneNum	TelNumType	optional
Affiliation	id	IdType	required

A.4.21 321-Questioner

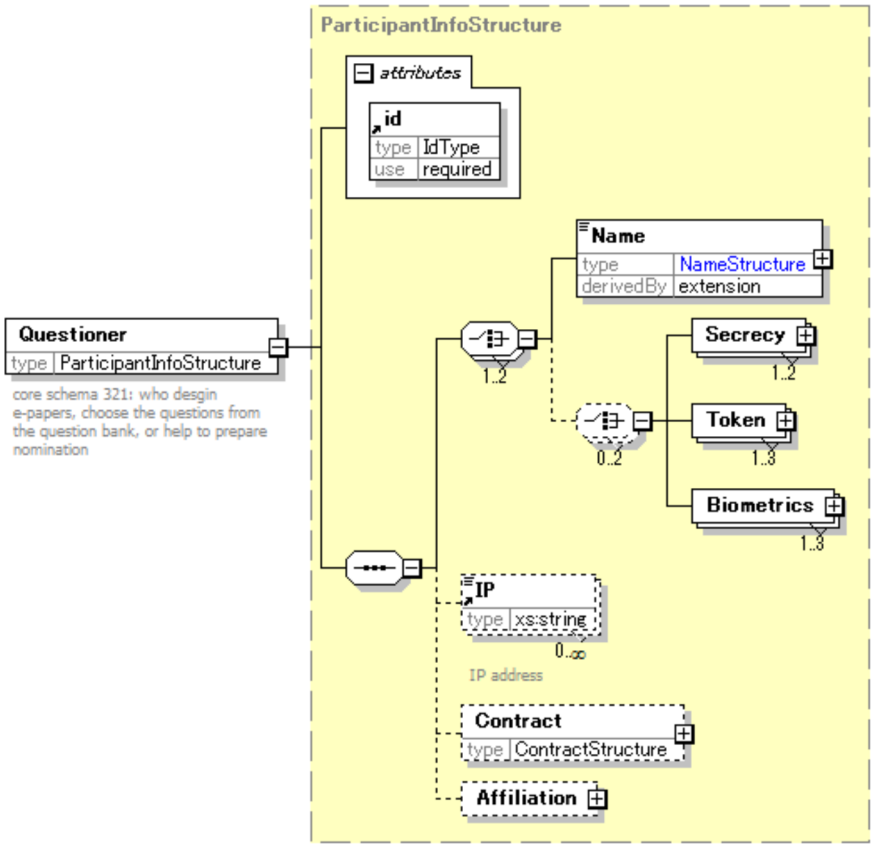

Figure A.37: Questioner diagram.

Fig. A.37 shows the schema about **Questioner**. This schema is used to define the questioner information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.4.22 322-Respondent

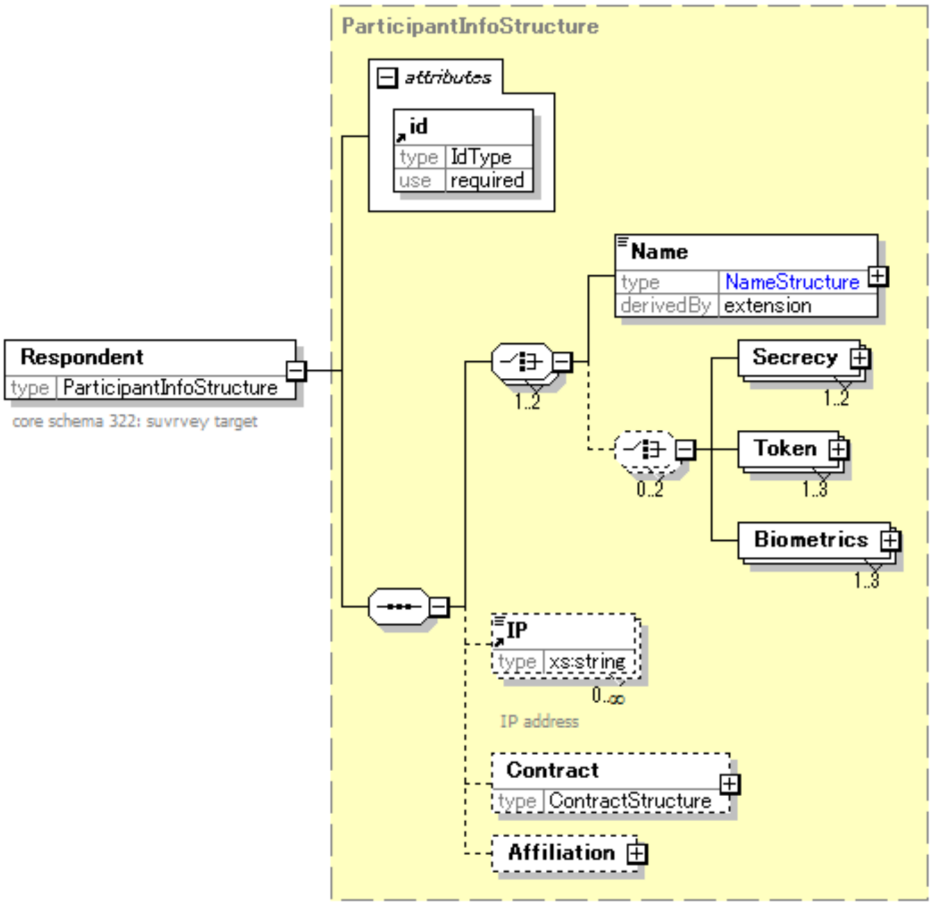

Figure A.38: Respondent diagram.

Fig. A.38 shows the schema about **Respondent**. This schema is used to define the respondent information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.4.23 323-Analyst

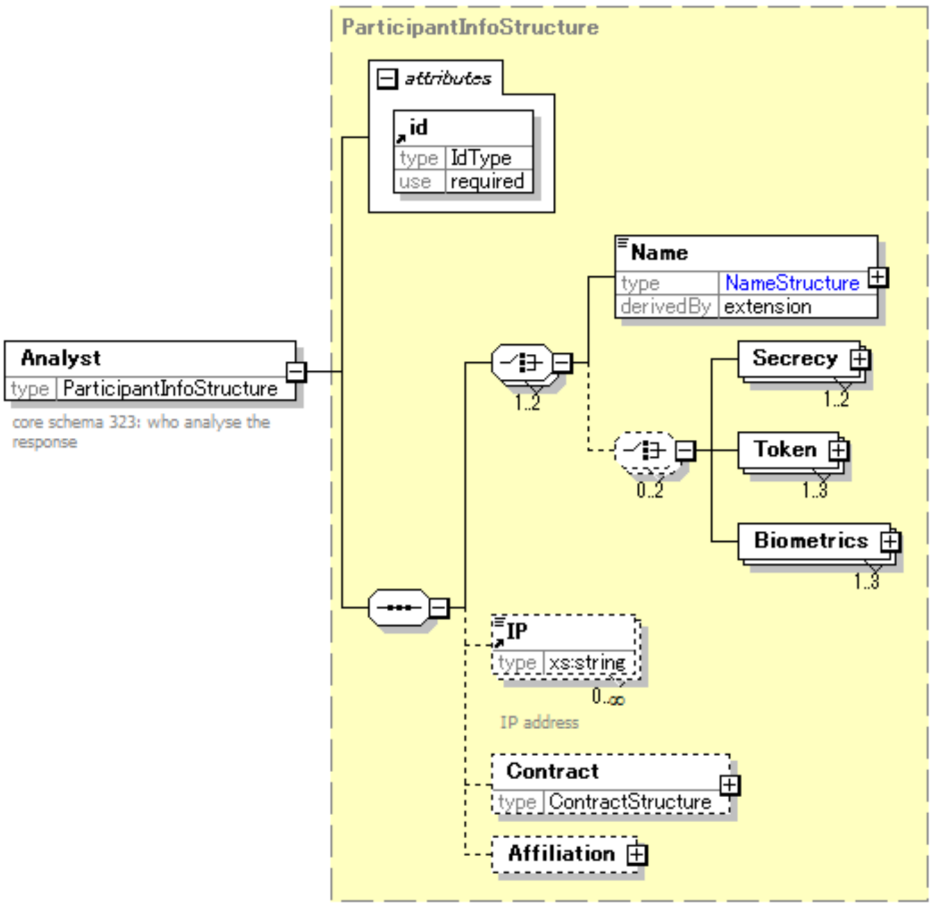

Figure A.39: Analyst diagram.

Fig. A.39 shows the schema about **Analyst**. This schema is used to define the analyst information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.4.24 324-Monitor

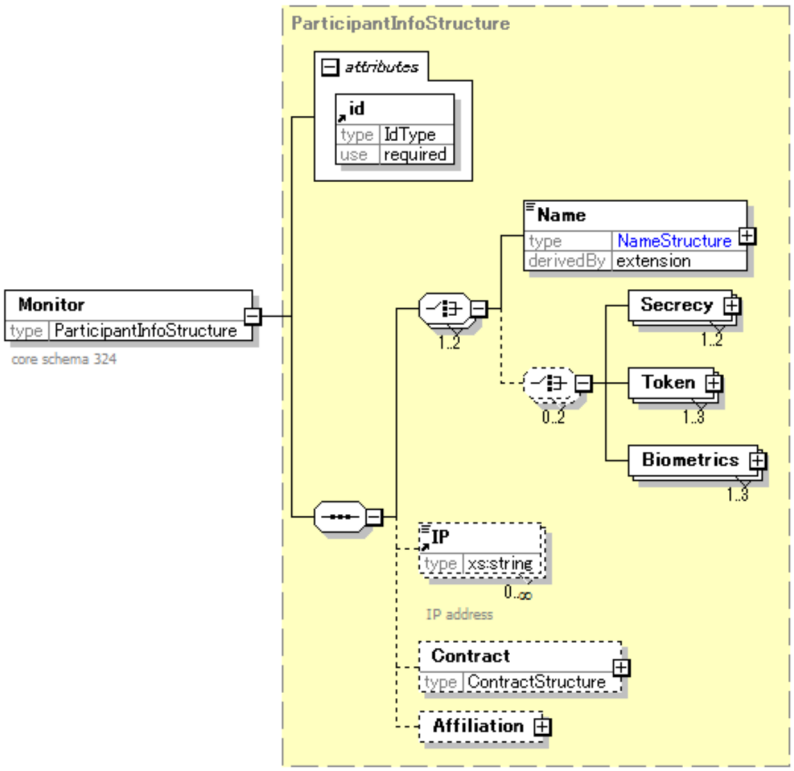

Figure A.40: Monitor diagram.

Fig. A.40 shows the schema about **Monitor**. This schema is used to define the monitor information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.4.25 330-Export

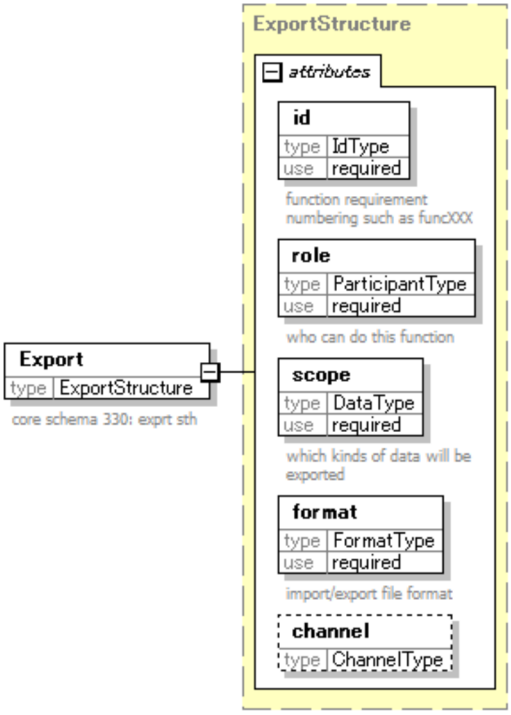

Figure A.41: Export diagram.

Element	Attribute	Type	Use
Export	id	IdType	required
	role	ParticipantType	required
	scope	DataType	required
	format	FormatType	required
	channel	ChannelType	optional

Fig. A.41 shows the schema about **Export**. This schema is used to define the export function. A participant can export a kind of data by a format, sometimes by a channel. There are some situations to use this function as follows. The *channel* attribute is set for extension of QSL that is not only for web system but also a machine. There is a list about different participants who can export what kinds of data during which kinds of phases as follows.

Participant	Phase	Data	Format	Channel
questioner	all	paper &setting, paper, setting	all	web link
respondent	after submitting	own responses	all	web link
analyst	after submitting	response, result	all	-
marker	after submitting	response, result	all	-
auditor	after submitting	response, result	all	-

A.4.26 331-Import

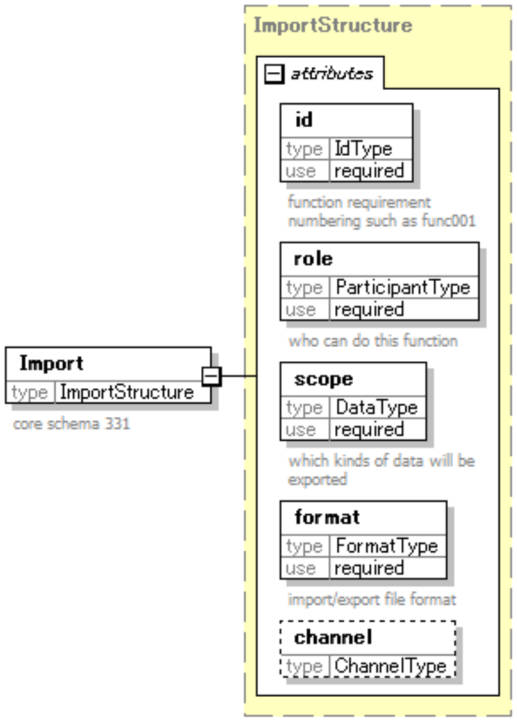

Figure A.42: Import diagram.

Element	Attribute	Type	Use
Import	id	IdType	required
	role	ParticipantType	required
	scope	DataType	required
	format	FormatType	required
	channel	ChannelType	optional

Fig. A.42 shows the schema about **Import**. This schema is used to define the import function. A participant can import a kind of data by a format, sometimes by a channel. There is a list about different participants who can import what kinds of data during which kinds of phases as follows.

Participant	Phase	Data	Format
questioner	setting	paper &setting, paper, setting	all
respondent	submitting	own responses	all
analyst	analyzing	response, result	all
marker	marking	response	all
auditor	auditing	response, result	all

A.4.27 332-Launch

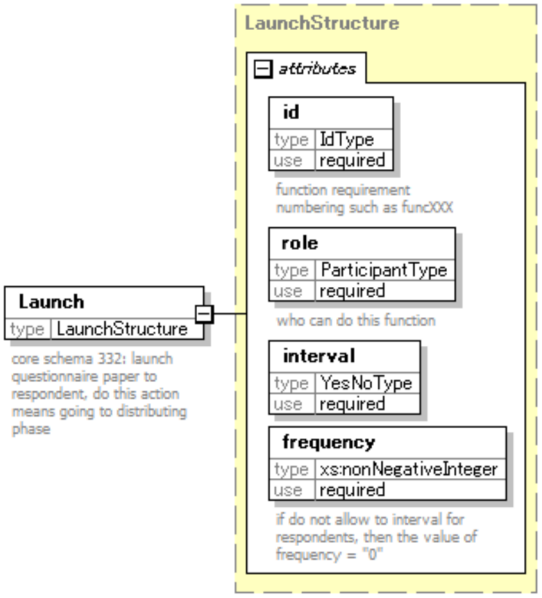

Figure A.43: Launch diagram.

Fig. A.43 shows the schema about **Launch**. This schema is used to define the launch function. Usually, a questioner can launch a paper into an active state for respondents. It is related with schema 333 **Stop** function.

Element	Attribute	Type	Use	Remark
Launch	id	IdType	required	identifier, value=funcXXX
	role	ParticipantType	required	who can do it
	interval	YesNoType	required	match with setting
	frequency	xs:nonNegativeInteger	required	value=0 means interval=no

A.4.28 333-Stop

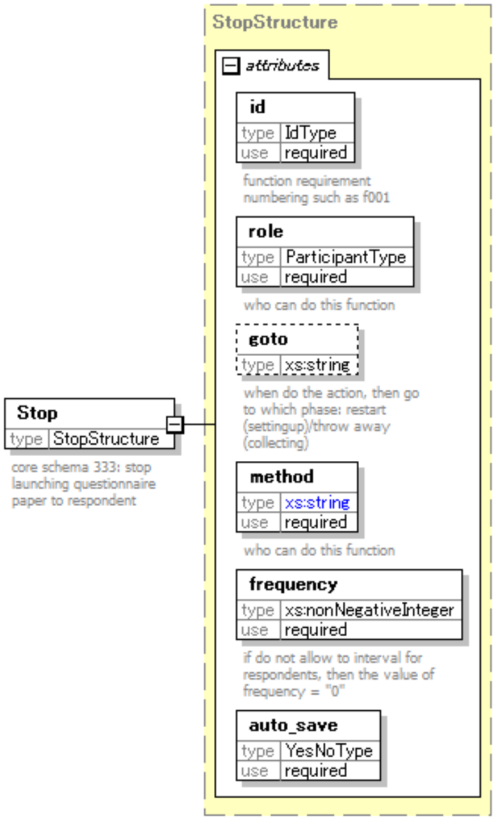

Figure A.44: Stop diagram.

Fig. A.44 shows the schema about **Stop**. This schema is used to define the stop launching function. Usually, a questioner can stop launching a paper into an inactive state for respondents, so that the respondents cannot answer anymore. The *goto* attribute is to specify the phase when doing this function. Usually, the restarted state is going to the setting up phase, and thrown-away state is going to collecting phase.

Element	Attribute	Type	Use	Remark
Stop	id	IdType	required	identifier, value=funcXXX
	role	ParticipantType	required	who can do it
	goto	PhaseType	optional	if action, then go to
	method	derived by xs:string	required	interval
	frequency	xs: nonNegativeInteger	required	value=0 means interval=no
	autoSave	YesNoType	required	-

A.4.29 334-Generate

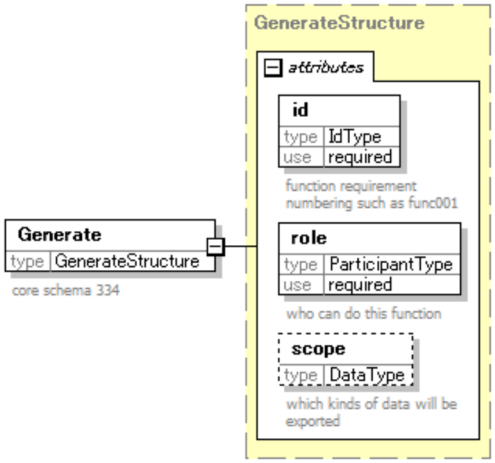

Figure A.45: Generate diagram.

Fig. A.45 shows the schema about **Generate**. This schema is used to define the generate function. Usually, a respondent login and register, so that he can get a registration ticket, usually in e-testing.

Element	Attribute	Type	Use	Remark
Generate	id	IdType	required	identifier, value=funcXXX
	role	ParticipantType	required	who can do it
	scope	DataType	optional	respondent registration ticket

A.4.30 335-Ping

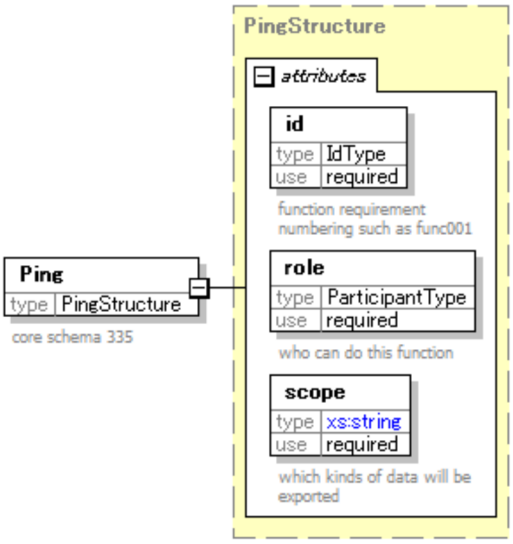

Figure A.46: Ping diagram.

Fig. A.46 shows the schema about **Ping**. This schema is used to define the ping function. Usually, a monitor to monitor the respondents ' IP addresses.

Element	Attribute	Type	Use	Remark
Stop	id	IdType	required	identifier, value=funcXXX
	role	ParticipantType	required	who can do it
	scope	DataType	required	IP

A.4.31 336-Integrate

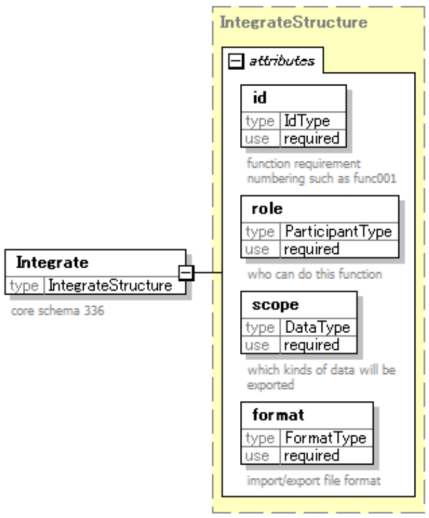

Figure A.47: Integrate diagram.

Fig. A.47 shows the schema about **Integrate**. This schema is used to define the integrate function. Usually, an analyst can integrate the responses into a zip or qsl file and export it.

Element	Attribute	Type	Use	Remark
Stop	id	IdType	required	identifier, value=funcXXX
	role	ParticipantType	required	who can do it
	scope	DataType	required	response
	format	FormatType	required	zip

A.4.32 337-Remind

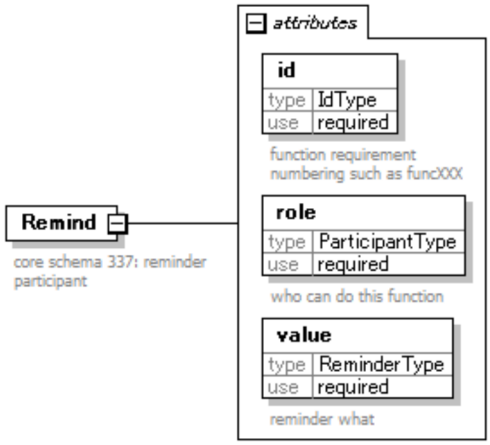

Figure A.48: Remind diagram.

Fig. A.48 shows the schema about **Remind**. This schema is used to define the remind function. The questioner can remind respondents that they have not finished answering or submitting yet with a notification mails.

Element	Attribute	Type	Use	Remark
Stop	id	IdType	required	identifier, value=funcXXX
	role	ParticipantType	required	who can do it
	value	ReminderType	required	notify data values

A.4.33 340-Server

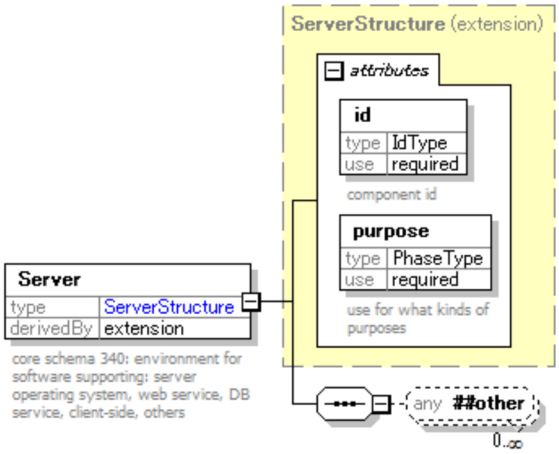

Figure A.49: Server diagram.

Fig. A.49 shows the schema about **Server**. This schema is used to specify server information for environment. This schema is based on the extension of complex type **ServerStructure**. We designed it to provide an interface.

Element	Attribute	Type	Use	Remark
Server	id	IdType	required	identifier
	purpose	PhaseType	required	phase

A.4.34 341-Field

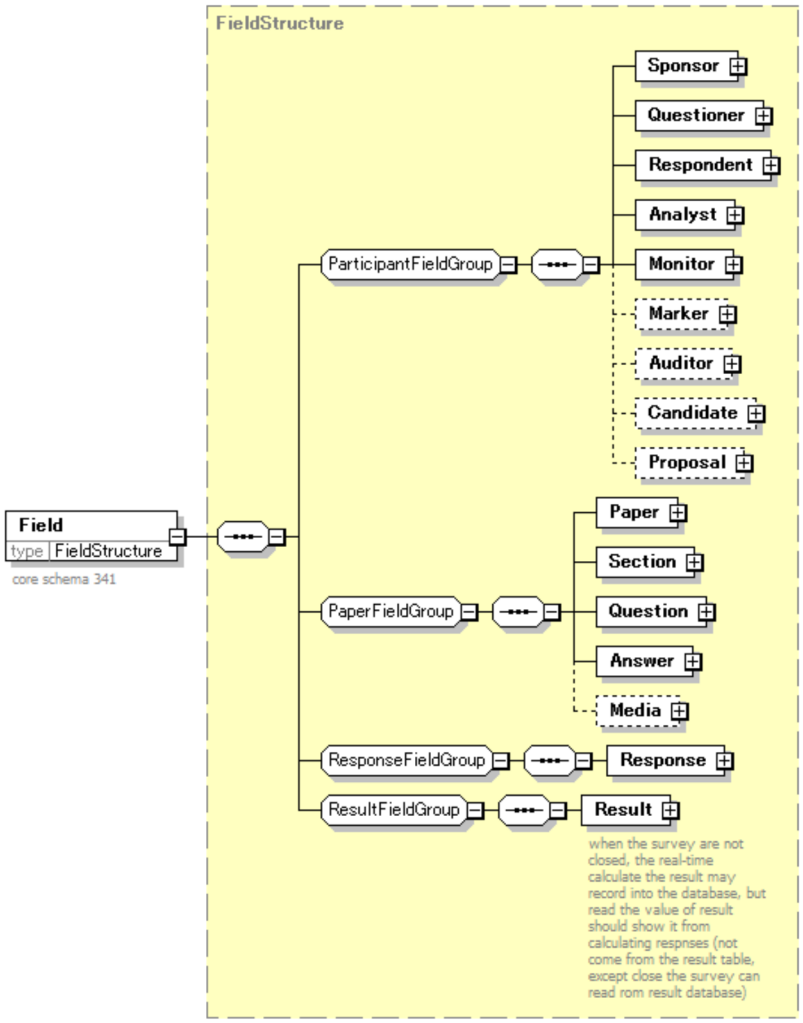

Figure A.50: Field diagram.

Fig. A.50 shows the schema about **Field**. This schema is used to specify database fields for each table in database. This schema is based on the complex type **FieldStructure**.

Element	Group	Remark
Field	ParticipantFieldGroup	to create tables for each kind of roles
	PaperFieldGroup	to create tables for its child elements
	ResponseFieldGroup	-
	ResultFieldGroup	-

- **Fields for Participant**

  Fig. A.51 is used to specify the fields for each role of participant in database. According to the phases and different services/ systems, the optional elements may be created in database. All the roles have same attributes, so we

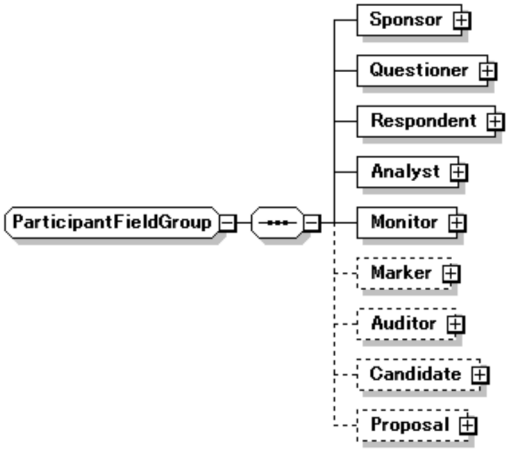

Figure A.51: Participant field group diagram.

defined the attribute group named "participantInfoAttriGroup." We give an example about **Sponsor** element, others can be referred to this schema as following in Fig. A.52.

Att.group	Attribute	Type	Use	Remark
participantInfo	id	FieldType	required	identifier
	familyName	FieldType	optional	family name
	FirstName	FieldType	optional	first name
	gender	FieldType	optional	gender
	nation	FieldType	optional	nation
	mail	FieldType	optional	e-mail
	telephone	FieldType	optional	telephone
	address1	FieldType	optional	detailed address info
	address2	FieldType	optional	city, state/ province
	address3	FieldType	optional	country

All the roles are used the attributes belonging to this attribute group. Except the **Proposer** element, it must to direct to the **Candidate** (nominee) (see in Fig. A.53).

- **Fields for Paper**

  Fig. A.54 is used to specify the fields for paper, section, question, answer, and media in database. If the paper is set with media, then the users should create table for media in database. The 5 elements are in a sequence order.

  Firstly, we defined the **Paper** element in Fig. A.55.

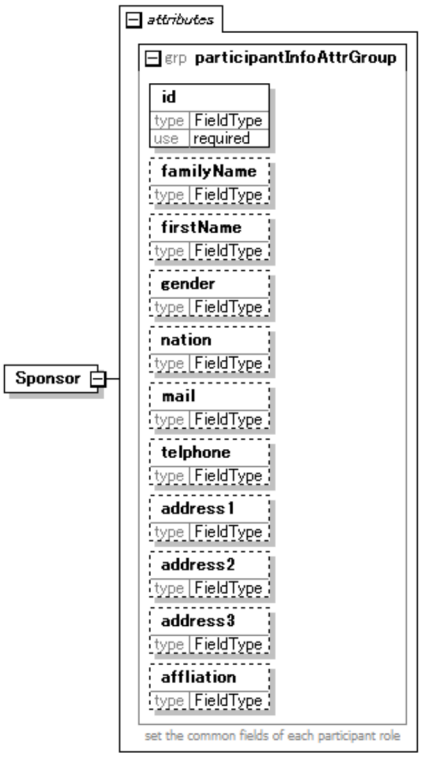

Figure A.52: Sponsor diagram.

Element	Attribute	Type	Use	Remark
Paper	id	FieldType	required	identifier
	title	FieldType	required	paper title
	description	FieldType	optional	paper subtitle
	conclusion	FieldType	optional	thanks page
	break	FieldType	optional	page break
	mediaId	FieldType	optional	if insert media
	situation	FieldType	required	active state of paper
	time	FieldType	optional	state time

This schema is used to specify fields of paper table in the database. **Section**, **Question**, and **Answer** elements refer to **Paper**.

- **Fields for Response**

  Fig. A.56 is used to specify the fields for respondents' responses in database. The child element Response is created as a table in the database. Fig. A.57 is the schema that is used to specify fields of response table in database. One row data is a respondent to answer a question.

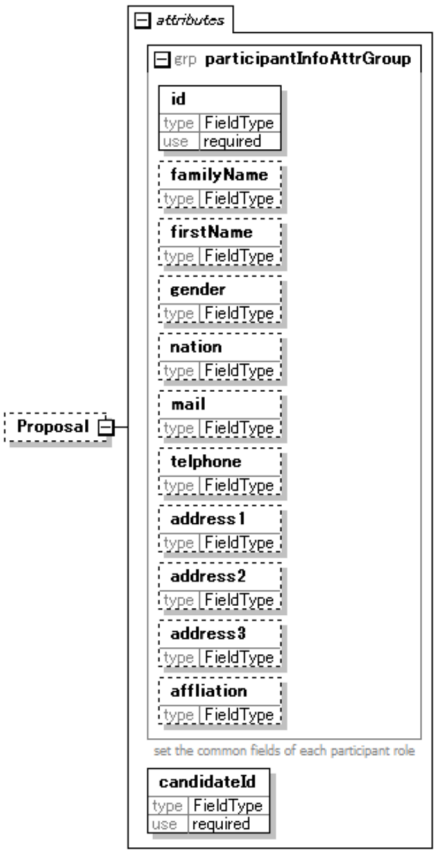

Figure A.53: Proposer diagram.

Element	Attribute	Type	Use	Remark
Response	id	FieldType	required	identifier
	content	FieldType	required	response content for a question
	respondentId	FieldType	required	respondent identifier
	questionId	FieldType	required	question identifier
	mediaId	FieldType	optional	media identifier
	time	FieldType	optional	time to answer a question

- **Fields for Result**

  Fig. A.58 is used to specify the fields for analyzed responses as a result in database. The child element **Result** is created as a table in the database. When the survey are not closed, the real-time calculate the result may record into the database, but read the value of result should show it from calculating responses (not come from the result table, except close the survey can read rom result database). Fig. A.59 is the schema that is used to specify fields of result table in database. One row data is a total result for a question.

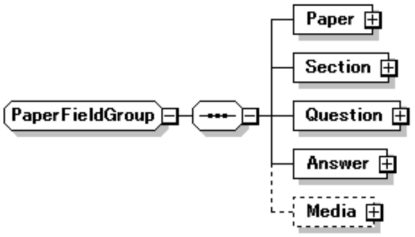

Figure A.54: Paper field group diagram.

Element	Attribute	Type	Use	Remark
Result	id	FieldType	required	identifier
	content	FieldType	required	response content for a question
	questionId	FieldType	required	question identifier
	time	FieldType	optional	time to answer a question

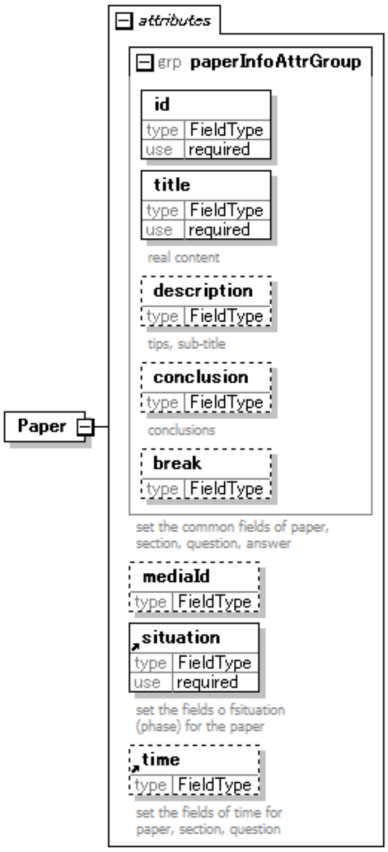

Figure A.55: Paper diagram.

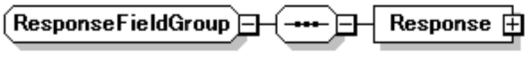

Figure A.56: Response field group diagram.

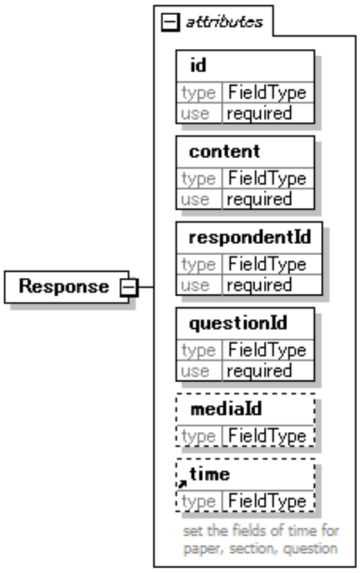

Figure A.57: Response diagram.

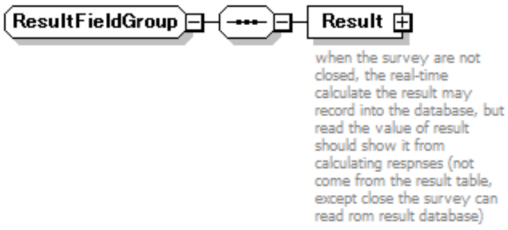

Figure A.58: Result field group diagram.

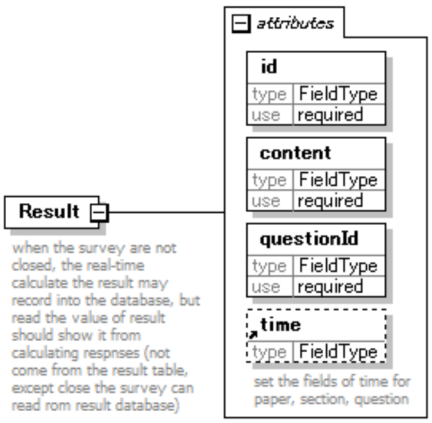

Figure A.59: Result diagram.

A.4.35 342-Gateway

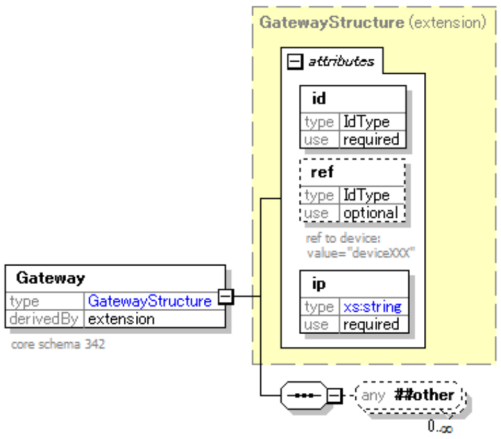

Figure A.60: Gateway diagram.

Fig. A.60 shows the schema about **Gateway**. This schema is used to specify network of environment, which should refer to corresponding devices, and record its IP. We designed it to provide its an interface for extension. Maybe the name of this tag is not correctly ("network" is better). The ip attribute is restricted its value in a mode of "[0-2](3). [0-9](3). [0-9](3). [0-9](3)" to limit. This schema is based on complex type of **GatewayStructure**.

Element	Attribute	Type	Use	Remark
Gateway	id	IdType	required	identifier
	ref	IdType	optional	refer to device identifier
	ip	derived by xs:string	required	IP address value

A.4.36 343-Interface

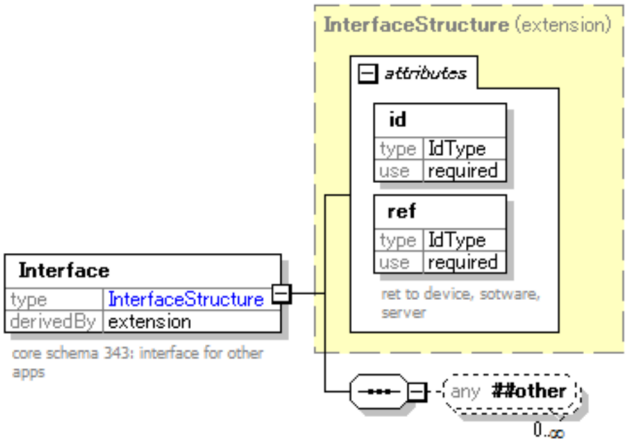

Figure A.61: Interface diagram.

Fig. A.61 shows the schema about **Interface**. This schema is used to specify interface of environment, which should refer to corresponding devices, software, or servers. We designed it to provide an interface for extension. This schema is based on complex type of **InterfaceStructure**.

Element	Attribute	Type	Use	Remark
Interface	id	IdType	required	identifier
	ref	IdType	optional	refer to device/software/server

A.4.37 344-Device

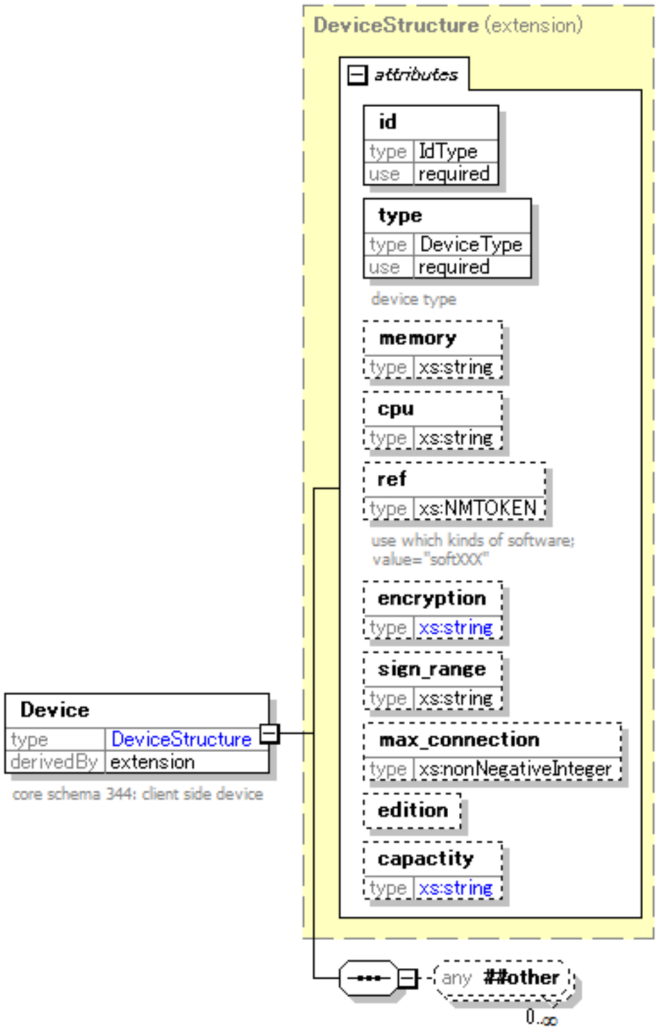

Figure A.62: Device diagram.

Fig. A.62 shows the schema about **Device**. This schema is used to specify device of environment. We designed it to provide an interface for extension. It is must to explain that there are 3 attribute groups for different devices. If the *type* attribute value is PC, then to select attributes in "PCattrGroup." This schema is based on complex type of **DeviceStructure**.

Attr.group	Attribute	Type	Use	Remark
-	id	IdType	required	identifier
-	type	DeviceType	optional	device type
PC	memory	xs:string	optional	8GB
	cpu	xs:string	optional	inter core i3
-	ref	xs:NMTOKEN	optional	r software identifier
access point	encryption	derived by xs:string	optional	WEP/WPA/others
	signRange	xs:string	optional	sign range
	maxConnection	xs:nonNegativeInteger	optional	max number
usb	edition	derived by xs:string	optional	2.0/3.0
	capacity	derived by xs:string	optional	1GB

A.4.38 345-Software

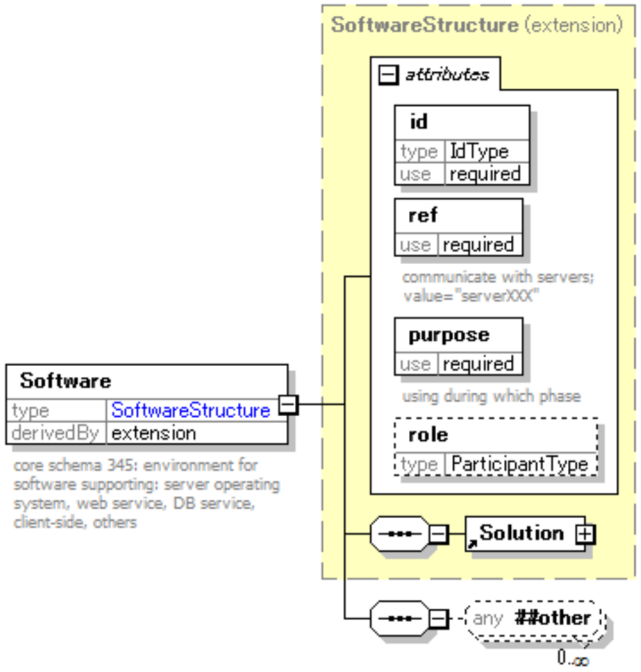

Figure A.63: Software diagram.

Fig. A.63 shows the schema about **Software**. This schema is used to specify software of environment. We designed it to provide the other interface for extension. This schema is based on complex type of **SoftwareStructure**.

Element	Attribute	Type	Use	Remark
Software	id	IdType	required	identifier
	ref	IdType	required	refer to server identifier
	purpose	derived by PhaseType	required	phase/ database/os
	role	ParticipantType	optional	-

A.4.39 350-Anonymity

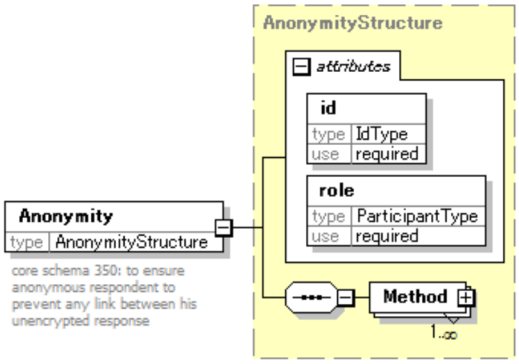

Figure A.64: Anonymity diagram.

Element	Attribute	Type	Use	Remark
Anonymity	id	IdType	required	identifier for setting
	role	ParticipantType	required	usually for respondents

Fig. A.64 shows the schema about **Anonymity**. This schema is used to specify anonymous respondents to prevent any link between unencrypted responses. This schema is based on the complex type **AnonymityStructure**. As to its child element **Method** (in Fig. A.65):

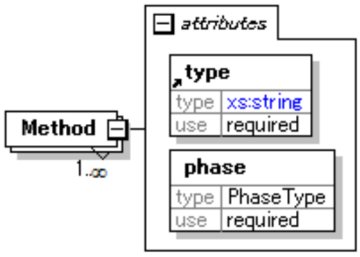

Figure A.65: Method diagram.

Element	Attribute	Type	Use	Remark
Method	type	xs:string	required	methods for anonymity
	phase	PhaseType	required	submitting, and registering

This schema is used to specify anonymous methods. You can use lots of methods for a system. The **Method** type is a xs:string enumeration facets, the value is limited as randomized auth token, blinded e-papers, blinded auth token, separation of duty, homomorphic secret, mix net, homomorphic encryption, HSM. As the values' meaning and the corresponding phase are listed below.

Method phase	Method type
Setting up	randomized auth token
Submitting	blind e-papers
	blinded auth token
	separation of duty
	homomorphic secret
Counting	mix net
	homomorphic encryption
	HSM

Blinded authenticated token: e-voting system should provide to digitally authenticate a message without knowing the content of the message. The respondent sends a blinded anonymous authentication token (instead of the blinded e-paper) to the validator together with some identification and authentication data. The respondent receives a digital signature from the validator on this blinded token. In the next step, the respondent computes the value for the signed authentication token and sends his data together with his e-paper to the tallier, which accepts the respondent because of the digitally signed authentication token.

Separation of duty: a separation of duty approach (also works with at least two submitting servers). One inspecting the right to submit and another one storing the eligible e-papers. The respondent authenticates himself to the first server. In case that he has the right to submit, he receives a random number generated by this first server. This random number is also sent to the second server but without any information about the respondent ID. Now the respondent uses this random number to authenticate himself as an eligible respondent to the second submitting server. Again this second server can only check whether an eligible respondent sent the e-paper but not who.

Mix Net: a mix-net for anonymity as a cryptographic alternative to an anonymous channel. It secures who is communicating with whom and it secures the content of the transferred messages.

HSM: a Hardware Security Module (HSM), which is a tamper-resistant or at least tamper-evident hardware component that can securely generate and store long-term secrets for use in cryptography. Generally, it is used to generate a digital key pair without revealing the private key. It can be a function that takes as the input the encrypted e-papers and returns as output the decrypted result, while the decrypted responses are not revealed.

A.4.40 351-Authentication

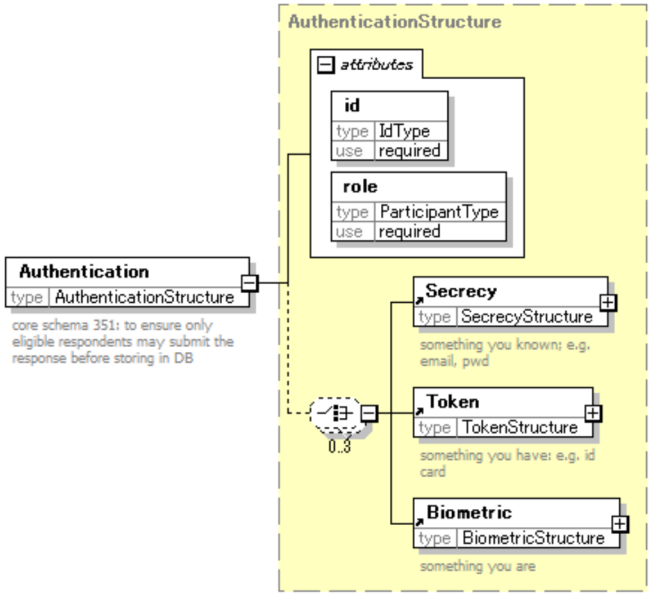

Figure A.66: Authentication diagram.

Fig. A.66 shows the schema about **Authentication**. This schema is used to specify authentication to ensure only eligible respondents can submit the responses before storing in database. It can also validate other roles of participants. This schema is based on the complex type **AuthenticationStructure**. There are 3 required elements but in a choice order. There are following possibilities to verify the users' identification.

Method	Child Element	Remark
sth. you known	Secrecy	-
sth. you have	Token	-
sth. you are	Biometric	-
duplicated verification	Secrecy & Token	usually used in e-t to check the token of respondents and let login in with secrecy
	Token & Biometric	usually used in e-v to double check respondent's token and login by biometric

Tips: You can set lots of duplicated verification by using two elements.

- **Secrecy**

  Fig. A.67 shows the schema about **Secrecy**. This schema is used to specify secrecy (something you known) for authentication. This schema is based on the complex type **SecrecyStructure**. There are 2 required elements in a sequence order. There are 2 required child elements.

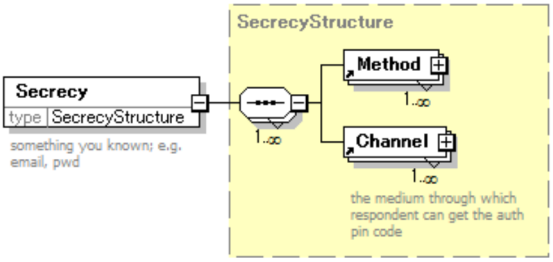

Figure A.67: Secrecy diagram.

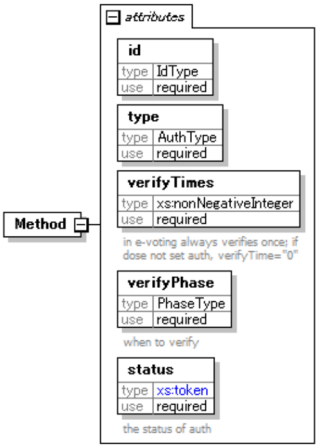

Figure A.68: Method diagram.

Fig. A.68 shows the schema about **Method**. This schema is used to specify secrecy methods. There are 5 required attributes. The status values are derived by restrictions, which are submitted, unsubmitted, issued, received, rejected, resubmitted, accepted, spoiled, exempted. **Method** type has values such as (for secrecy: password, randomized pwd, for token: ID card, job card, library card, roll card, for biometric: finger prints, iris, face recognition, DNA, others. **Channel** element: includes schema 354.

- **Token**

  Fig. A.69 shows the schema about **Token**. This schema is used to specify token (something you have) for authentication. This schema is based on the complex type **TokenStructure**. There are 2 required elements in a sequence order. There are 2 required child elements. **Method** and **Channel** (schema 354) are included and shared among the **Authentication**.

- **Biometric**

  Fig. A.70 shows the schema about **Biometric**. This schema is used to specify biometric (something you are) for authentication. This schema is based on the complex type **TokenStructure**. There are 2 required elements in a

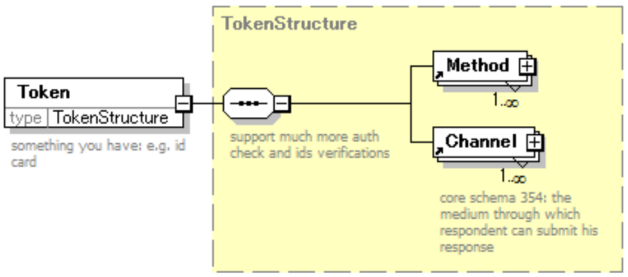

Figure A.69: Token diagram.

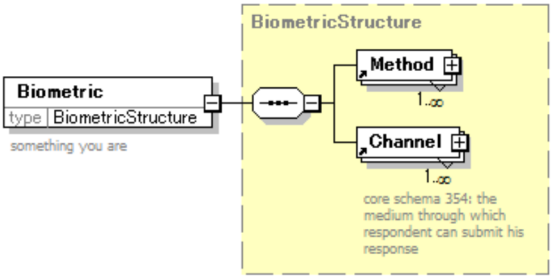

Figure A.70: Biometric diagram.

sequence order. There are 2 required child elements. **Method** and **Channel** (schema 354) are included and shared among the **Authentication**.

A.4.41 352-Authority

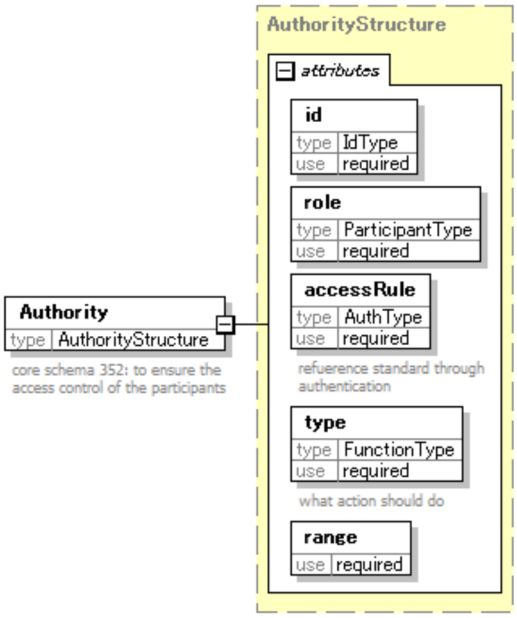

Figure A.71: Authority diagram.

Fig. A.71 shows the schema about **Authority**. This schema is used to specify participant who can access control to which phases and which functions, and which data. Except id attribute, the remaining 4 attributes belong to an attribute group named "roughAccessAttrType." This schema is based on the complex type **AuthorityStructure**.

Attribute	Type	Use	Remark
id	IdType	required	identifier
role	ParticipantType	required	to whom can access
accessRule	AuthType	required	
type	PhaseType	required	which phase
range	union: FunctionType, and DataType	required	which can operate

A.4.42 353-Seal

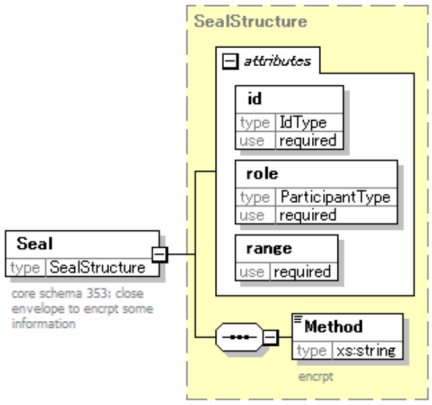

Figure A.72: Seal diagram.

Fig. A.72 shows the schema about **Seal**. This schema is used to specify to encrypt what kinds of data. This schema is based on the complex type **SealStructure**. The Method element is as interface for extension.

Element	Attribute	Type	Use	Remark
Seal	id	IdType	required	identifier
	role	ParticipantType	required	to whom can encrypt
	range	DataType	required	encrypt what

A.4.43 354-Channel

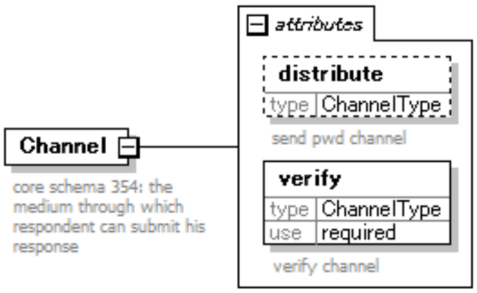

Figure A.73: Channel diagram.

Fig. A.73 shows the schema about **Channel**. This schema is used to specify the medium through by which respondent can submit his response.

Element	Attribute	Type	Use	Remark
Channel	distribute	ChannelType	optional	method for distributing
	verify	ChannelType	required	method

A.4.44 360-Section

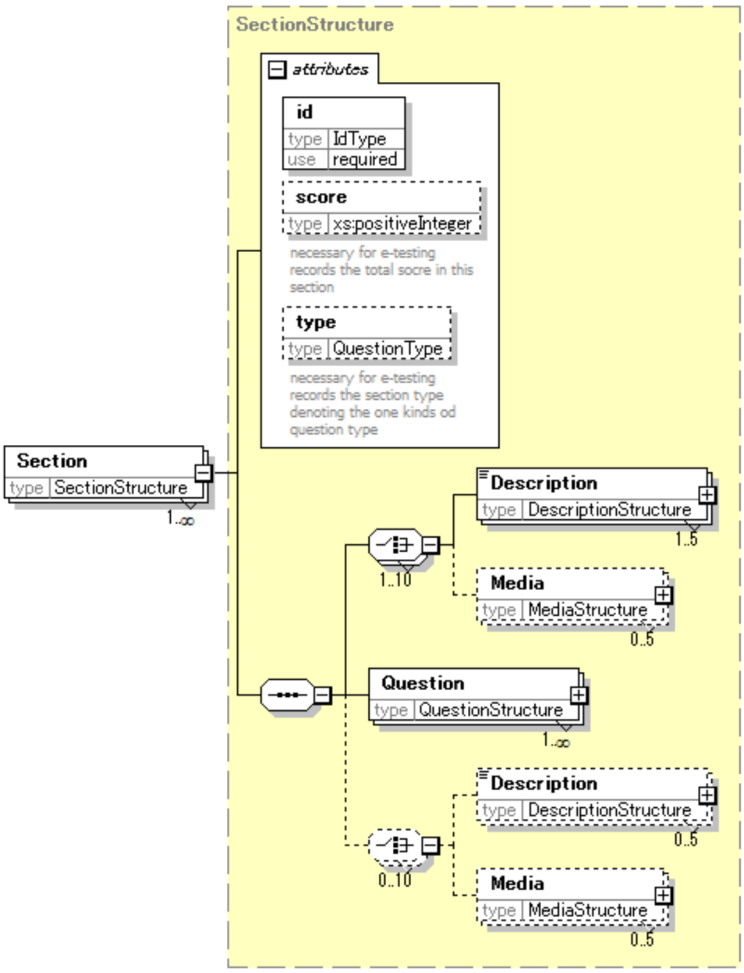

Figure A.74: Section diagram.

Fig. A.74 shows the schema about **Section**. This schema is used to specify section for a paper sheet. This schema is based on the complex type **Section-Structure**. There are 3 attributes listed in table, and a choice model to specify section title (required, no more than 5) and media (optional, no more than 5), and sequence model to specify questions. The *type* attribute is used to describe the question types in this section are same or not, usually used for e-testing.

Attribute	Type	Use	Remark
id	IdType	required	identifier
score	xs:positiveInteger	optional	section score for e-testing
type	QuestionType	optional	

A.4.45 361-Question

Fig. A.75 shows the schema about **Question**. This schema is used to specify question for a paper sheet. This schema is based on the complex type **Question-Structure**. There are 5 attributes listed in table, and a choice model to specify question title (required, no more than 5) and media (optional, no more than 5), and sequence model to specify answers. The *textAlign* is used for alignment of prefix and suffix (alignment relation between description of question and text area of answer)

Attribute	Type	Use	Remark
id	IdType	required	identifier; value: q001
type	QuestionType	required	if section type chosen, then question type is similar with that
isMandatory	YesNoType	required	necessary to answer
score	xs:positiveInteger	optional	question total score
textAlign	derived by xs:string	optional	

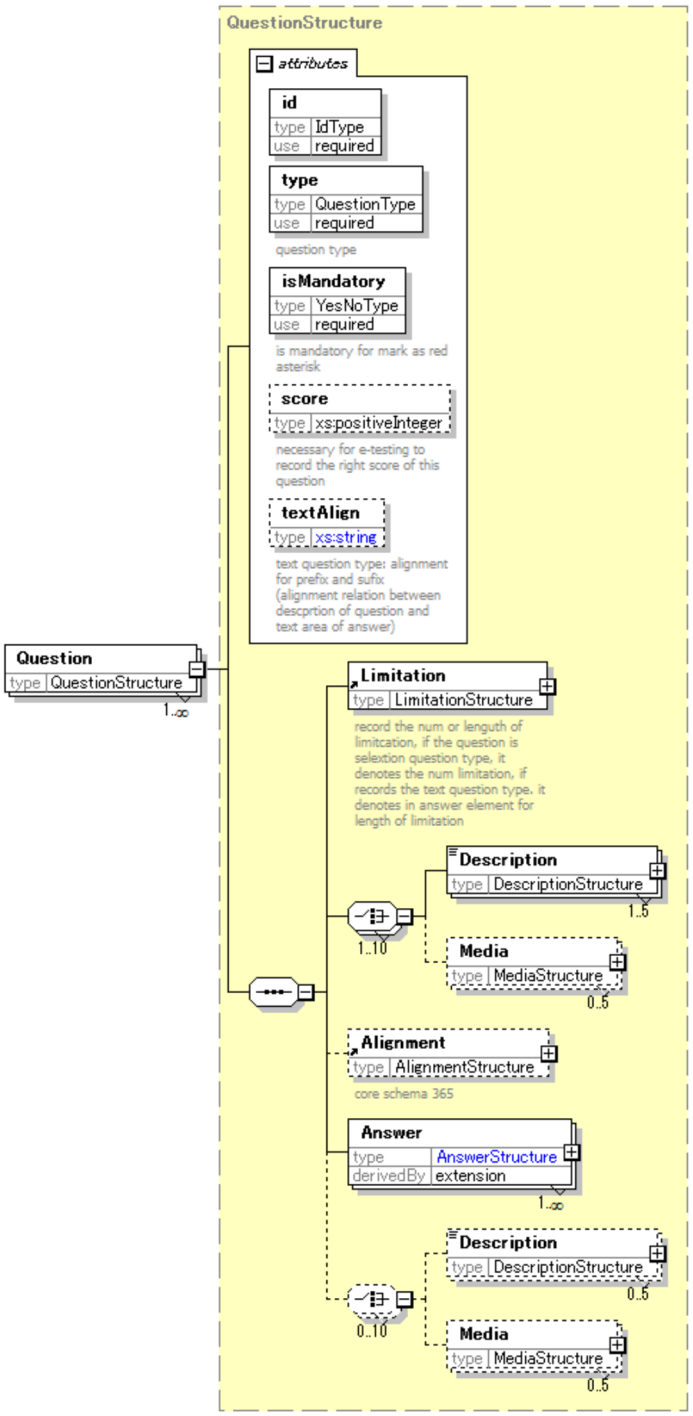

Figure A.75: Question diagram.

A.4.46 362-Answer

Fig. A.76 shows the schema about **Answer**. This schema is used to specify an answer. This schema is based on the complex type **AnswerStructure**. **Limitation** helps to limit the text characteristics, and options numbers. Formula (schema 540) helps to define the math, chemical, and other interfaces. *isDynamic2Next* is used to provide relationship of hierarchies of data such ad zip code and corresponding addresses.

Attribute	Type	Use	Remark
id	IdType	required	answer id: anXXX
isDefault	YesNoType	optional	sample answer for e-testing
isNAOption	YesNoType	optional	*isNAOption* exclusive
isOtherField	YesNoType	optional	*isOtherField* exclusive
Column.Grid	union:	optional	for 2D matrix
Row.Grid	xs:nonNegativeInteger	optional	suffix from 0
X.Grid	xs:token	optional	for 3D matrix
Y.Grid		optional	(matrix+drop-down list)
Z.Grid		optional	suffix from 0
weigtht	xs: double	optional	weight affects the results
score	xs:double	optional	answer score rules
defaultValue	xs:token	optional	text and ranking sample
isDragged	YesNoType	optional	for ranking
size	derived by xs:string	optional	text area size (pixel*pixel)
isDynamic2Next	YesNoType	optional	

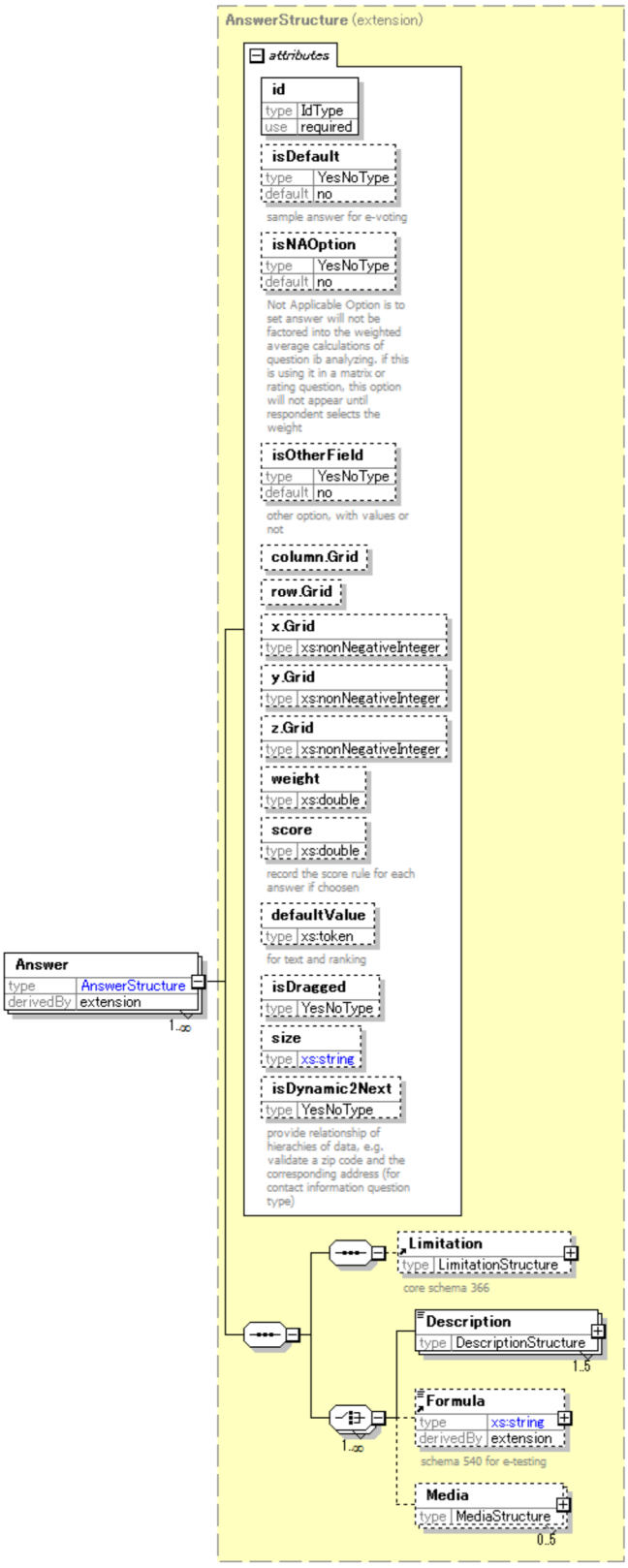

Figure A.76: Answer diagram.

A.4.47 363-Description

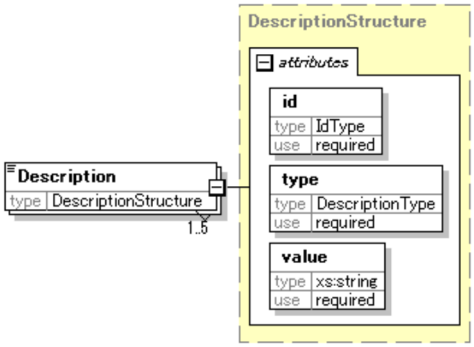

Figure A.77: Description diagram.

Fig. A.77 shows the schema about **Description**. This schema is used to specify contents for paper, section, question, and answer. It has 3 attributes. Usually, the description type chooses the title as its value. The description value should be stored in the database. This schema is based on the complex type **DescriptionStructure**.

Element	Attribute	Type	Use	Remark
Description	id	IdType	required	text identifier
	type	DescriptionType	required	distinguish title, subtitle, and summary
	value	xs:string	required	contents

A.4.48 364-Media

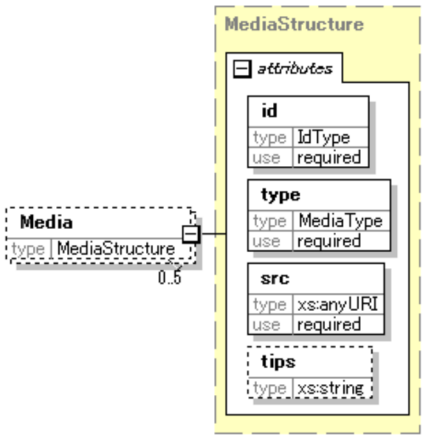

Figure A.78: Media diagram.

Fig. A.78 shows the schema about **Media**. This schema is used to specify contents for paper, section, question, and answer. It has 3 attributes. Usually, the description type chooses title as its value. The description value should be stored in database. This schema is based on the complex type **DescriptionStructure**.

Element	Attribute	Type	Use	Remark
Media	id	IdType	required	identifier
	type	MediaType	required	media file suffix
	src	xs:anyURI	required	roots
	tips	xs:string	required	mouse hovered the media

A.4.49 365-Alignment

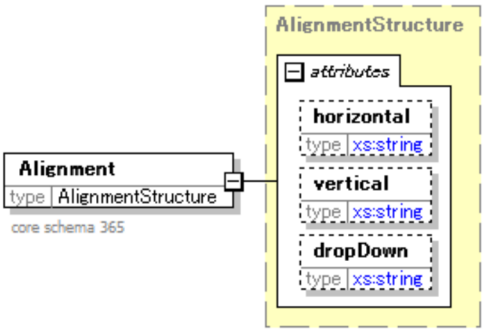

Figure A.79: Alignment diagram.

Fig. A.79 shows the schema about **Alignment**. This specified the arrangement of answer and questions. It is an optional child element of **Question**. Considering questioner could set arrangement of questions and answers (especially for selection question type). These attributes refer to xAML.

Element	Attribute	Type	Use	Remark
Alignment	horizontal	restriction of xs:token	optional	value: left, right, center, stretch
	vertical	restriction of xs:token	optional	value: center, stretch, top, bottom
	dropdown	restriction of xs:token	optional	value: listView, listBox

A.4.50 366-Limitation

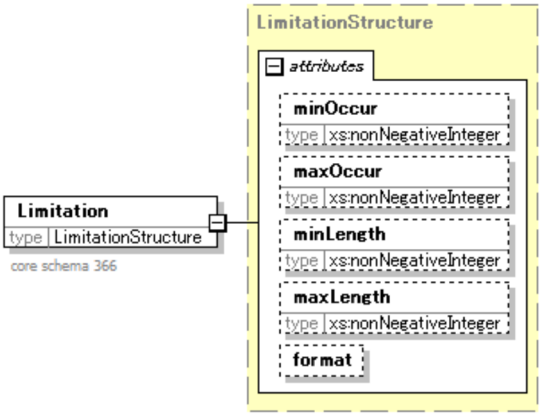

Figure A.80: Limitation diagram.

Fig. A.80 shows the schema about **Limitation**. This schema is used to specify limitation for questions and answers.

Element	Attribute	Type	Use
Limitation	minOccur	xs:nonNegativeInteger	optional
	maxOccur	xs:nonNegativeInteger	optional
	minLength	xs:nonNegativeInteger	optional
	maxLength	xs:nonNegativeInteger	optional
	format	union: FieldType, xs:anyURI,	optional
	format	EmailType, xs:tring	

A.4.51 370-Language

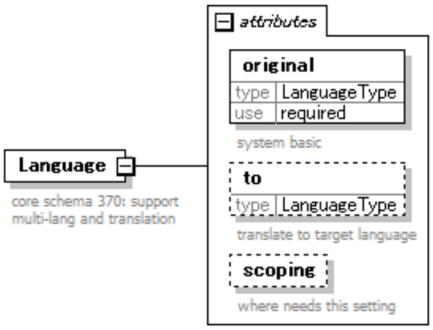

Figure A.81: Language diagram.

Fig. A.81 shows the schema about **Language**.

Element	Attribute	Type	Use	Remark
Language	original to scoping	LanguageType LanguageType Union: PaperType &DataType &SystemType	required optional optional	 translate to if need translate; value: system, paper, response, report

A.4.52 371-Time

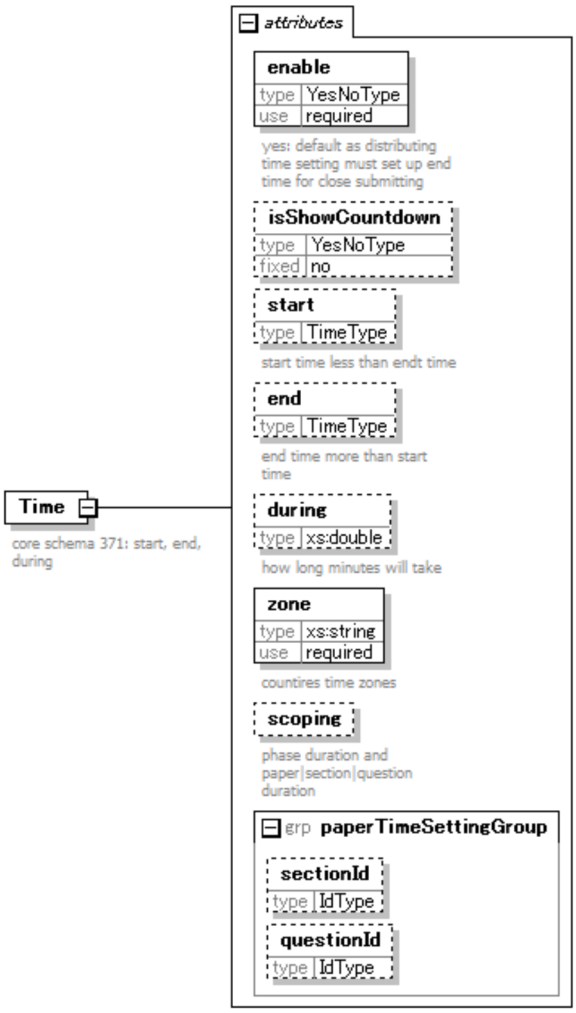

Figure A.82: Time diagram.

Fig. A.82 shows the schema about **Time**.

Element	Attribute	Type	Use	Remark
Time	enable	YesNoType	required	-
	isShowCountdown	YesNoType	optional	fixed value no
	zone	derived by xs:string	required	UTC+9
	start	TimeType	optional	during exclusive
	end	TimeType	optional	-
	during	xs:double	optional	start/end exclusive
	scoping	union: PhaseType &PaperType	optional	-
paperTime SettingGroup	sectionId	IdType	optional	-
	questionId	IdType	optional	-

A.4.53 372-Number

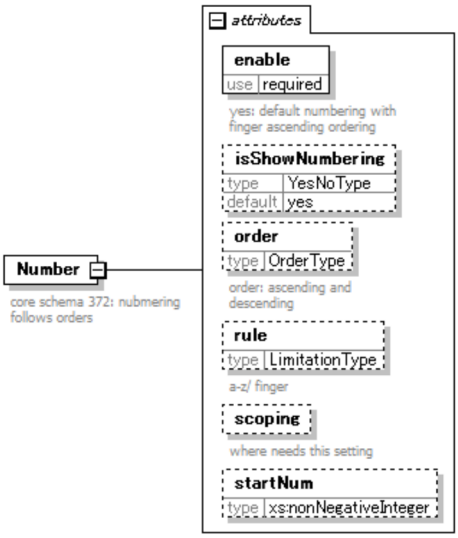

Figure A.83: Number diagram.

Element	Attribute	Type	Use
Number	enable	YesNoType	required
	isShowNumbering	YesNoType	optional
	order	OrderType	optional
	rule	LimtitationType	optional
	scoping	union: PaperType& DataType	optional
	startNum	xs:nonNegativeInteger	optional

Fig. A.83 shows the schema about **Number**. This schema is used to specify the ordering setting. The *isShowNumbering* attribute is to fixedly shown the fingers by an ascending order.

A.4.54 373-Quota

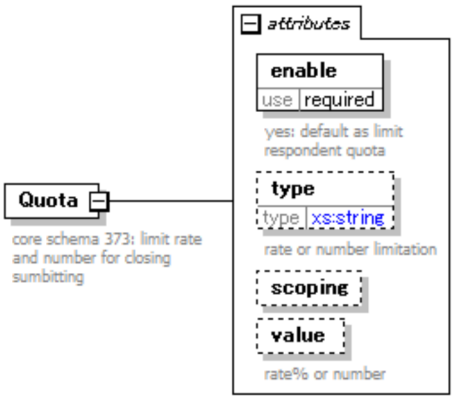

Figure A.84: Quota diagram.

Element	Attribute	Type	Use
Quota	enable	YesNoType	required
	type	derived by xs:string	optional
	scoping	union: ParticipantType&DataType	optional
	value	xs:string	optional

Fig. A.84 shows the schema about **Quota**. This schema is used to specify the quota limitation for setting the numbers of the respondent to access. There are two kinds of quota types to limit the respondents. It is the intersection relation with the authentication methods.

A.4.55 374-Response

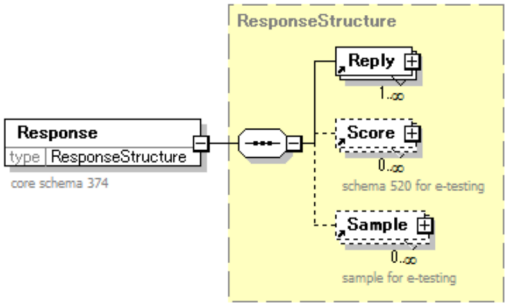

Figure A.85: Response diagram.

Fig. A.85 shows the schema about **Response**. This schema is used to specify response information recorded in the database. It has 3 children elements. This schema is based on the complex type **ResponseStructure**. If the users design an e-testing system or service, it should record the score (schema 520) and sample answer (schema 530). Fig. A.86 shows its child element named **Reply**.

Element	Attribute	Type	Use
Reply	id	IdType	required
	respondentId	IdType	required
	paperId	IdType	required
	sectionId	IdType	required
	questionId	IdType	required
	answerId	IdType	required
	value	xs:string	required

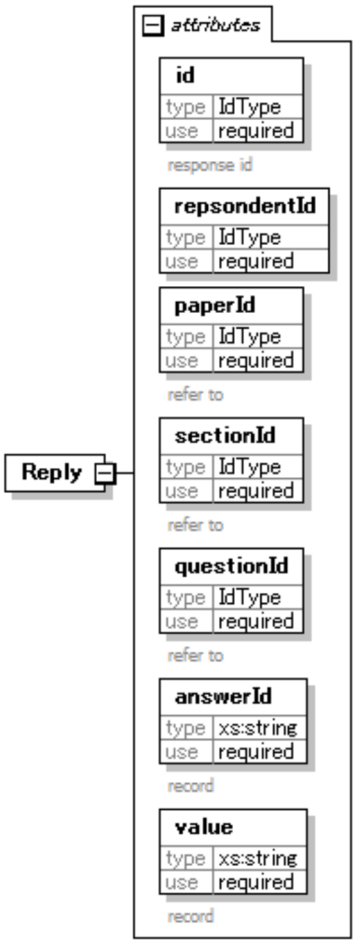

Figure A.86: Reply diagram.

A.4.56 375-Result

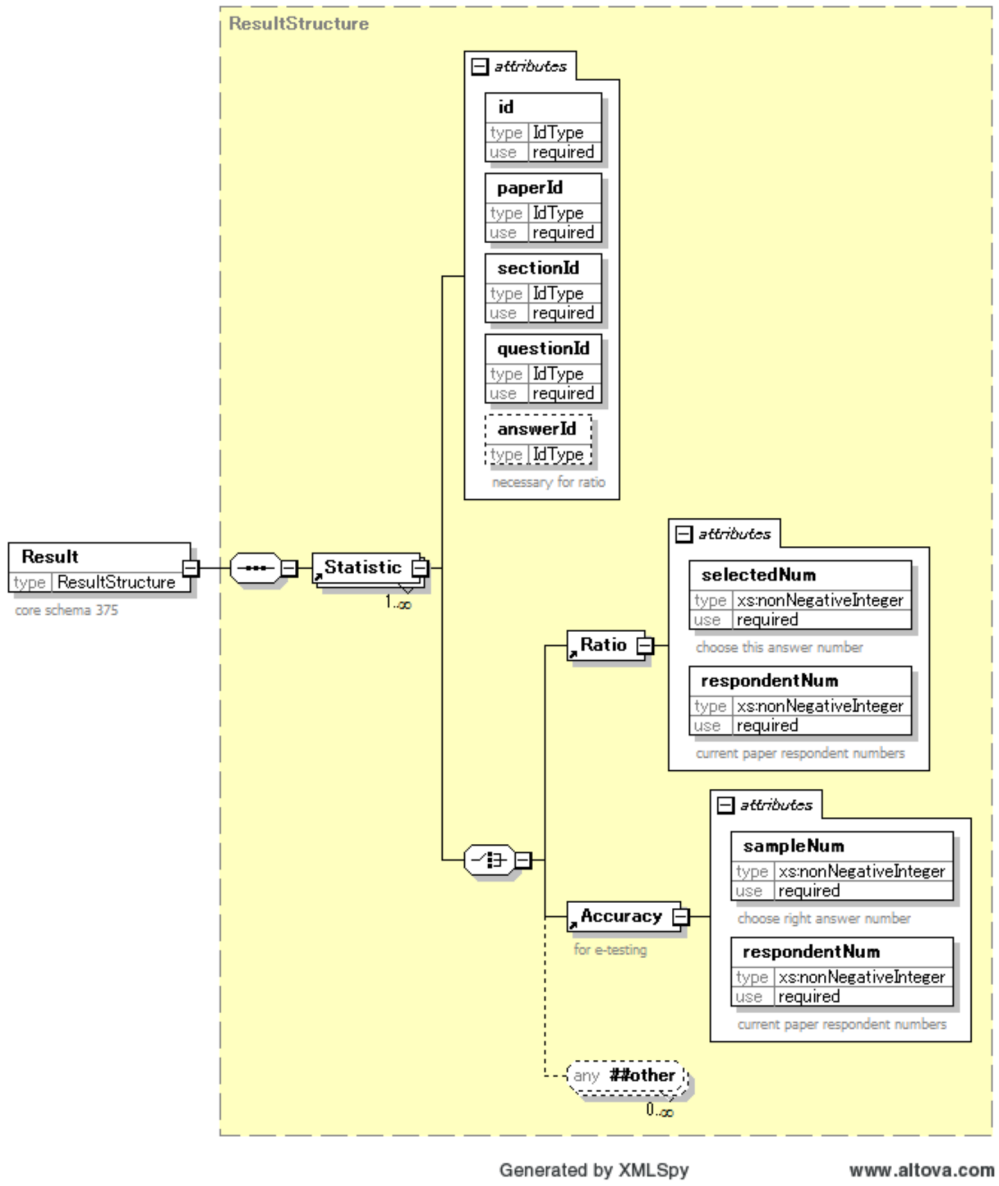

Figure A.87: Result diagram.

Element	Attribute	Type	Use
Statistic	id	IdType	required
	paperId	IdType	required
	sectionId	IdType	required
	questionId	IdType	required
	answerId	IdType	optional
Ratio	selectedNum	xs:nonNegativeInteger	required
	respondentNum	xs:nonNegativeInteger	required
Accuracy	sampleNum	xs:nonNegativeInteger	required
	respondentNum	xs:nonNegativeInteger	required

Fig. A.87 shows the schema about **Result**. **Result** schema is used to define the report data in details. Its child element **Statistic** used to specify a lot of report style. It determines by setting for result types. This provides the reference to detailed answers identifier and provides the **Ratio** to completeness and **Accuracy** for e-testing. Both child elements of statistic are in a choice order.

A.4.57 410-Logic

Element	Attribute	Type	Use
Route	id	IdType	required
	type	LogicType	required
Condition	paperId	IdType	optional
	sectionId	IdType	optional
	questionId	IdType	optional
	answerId	IdType	optional
	isChecked	YesNoType	optional
	relation	restriction of xs:token	optional
	score	xs:double	optional
Action	paperId	IdType	optional
	sectionId	IdType	optional
	questionId	IdType	optional
	answerId	IdType	optional
	paperId	IdType	optional
	isFilled	YesNoType	optional
	descriptionId	IdType	optional
	to	union: xs:anyURI &restriction of xs:token pattern:" $('pip[0-9]*'), $('ext[0-9]*')	optional

Fig. A.88 shows the schema about **Logic**. Schema **Logic** has a child element Route to record the routes of the logic from which **Condition** and go to which **Action**. Much more logic types and the templates for guiding how to use it in details are listed in Section A.6.

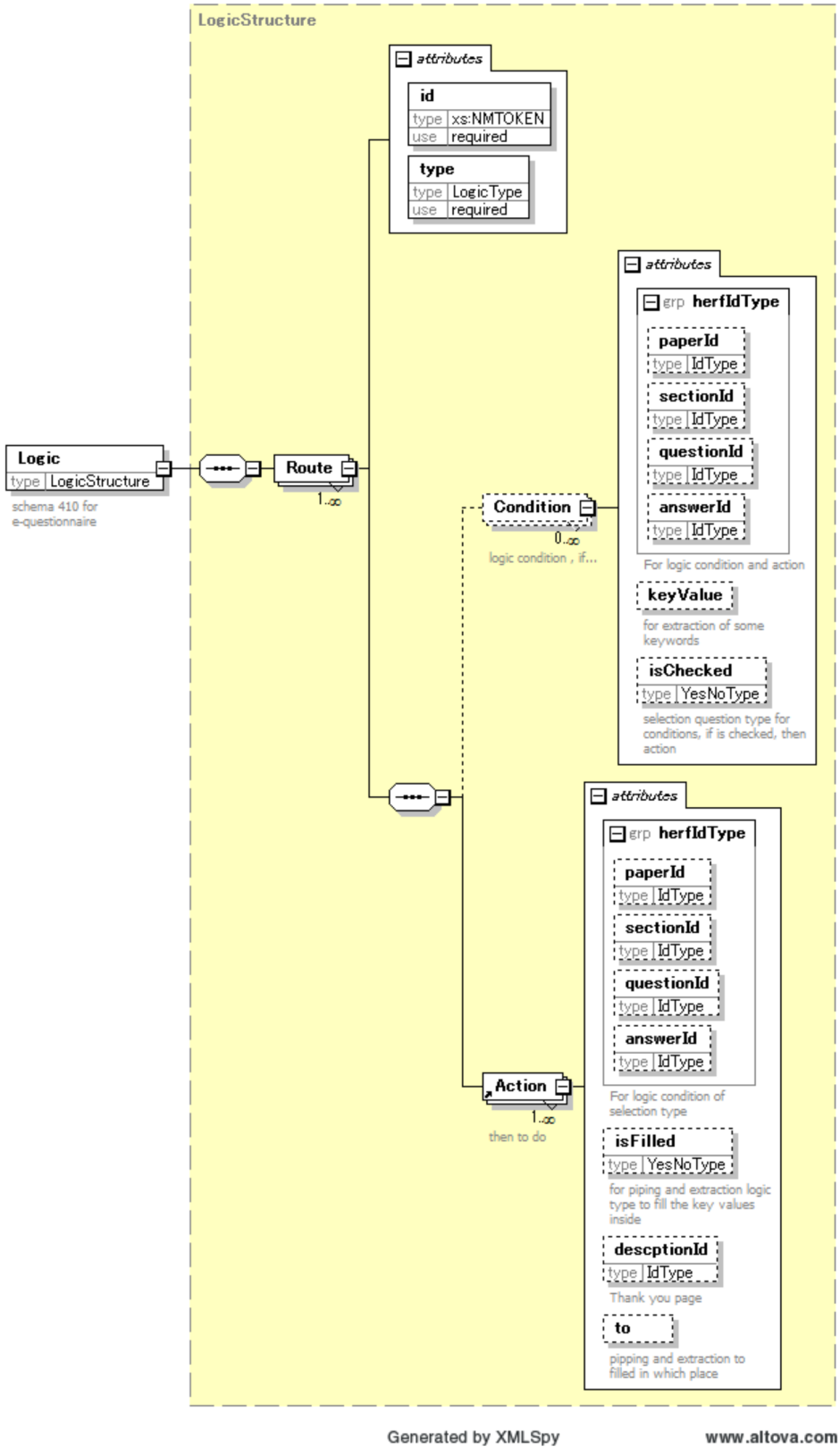

Figure A.88: Logic diagram.

A.4.58 510-Marking

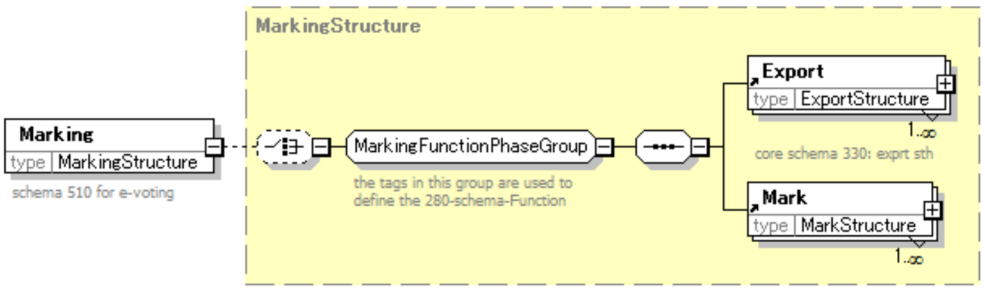

Figure A.89: Marking diagram.

Element	Group	Remark
Marking	MarkingFunctionPhaseGroup	belong to 280 schema

Fig. A.89 shows the schema about **Marking**. This schema is used to specify the phase to mark the responses in e-testing. This schema is based on the complex type **MarkingStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we do not define the settings during this phase. The element in the function group is explained in core schemas (schema 330), please refer to the corresponding schema definition. As the last child element, **Mark** is not listed in the schema file, because it has the duplicated meanings with other tags.

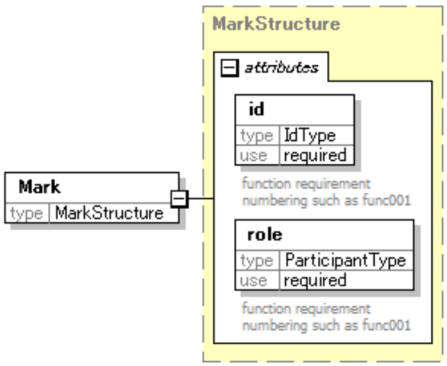

Figure A.90: Mark diagram.

Element	Attribute	Type	Use
Mark	id	IdType	required
	role	ParticipantType	required

Fig. A.90 shows the schema about **Mark**. This schema is used to define the mark function. The system should provide interface to mark response data to get a score.

A.4.59 520-Score

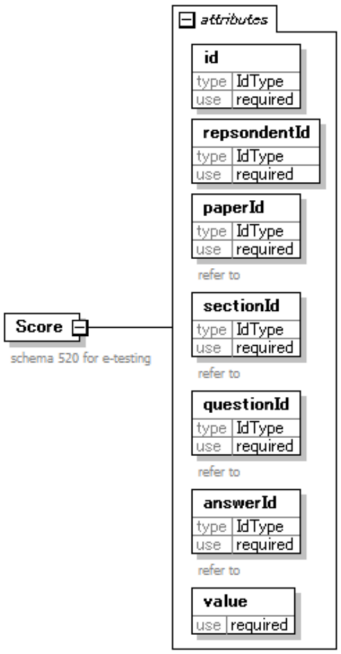

Figure A.91: Score diagram.

Element	Attribute	Type	Use
Score	id	IdType	required
	respondentId	IdType	required
	questionerId	IdType	required
	paperId	IdType	required
	sectionId	IdType	required
	questionId	IdType	required
	answerId	IdType	required
	value	union: xs:double&xs:token	required

Fig. A.91 shows the schema about **Score**. Schema 520 provides the specification for score data in details. As to the score setting rule is specified in paper sheet.

A.4.60 530-Sample

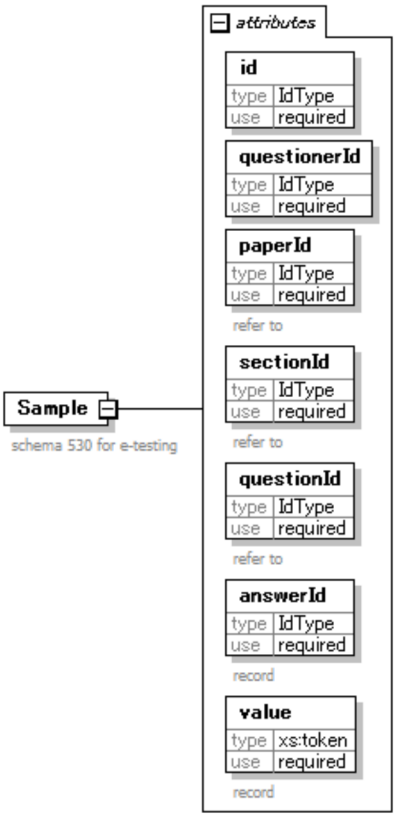

Figure A.92: Sample diagram.

Element	Attribute	Type	Use
Sample	id	IdType	required
	questionerId	IdType	required
	paperId	IdType	required
	sectionId	IdType	required
	questionId	IdType	required
	answerId	IdType	required
	value	xs:token	required

Fig. A.92 shows the schema about **Sample**. Schema 530 provides the specification for sample answer and its questioner.

A.4.61 540-Formula

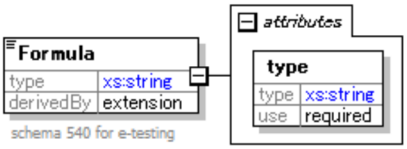

Figure A.93: Formula diagram.

Element	Attribute	Type	Use
Formula	type	derived by xs:string	required

Fig. A.93 shows the schema about **Formula**. Schema 540 provides the skeleton of specification for formula about fields of math, chemistry, physic, programing, biology, other.

A.4.62 550-Marker

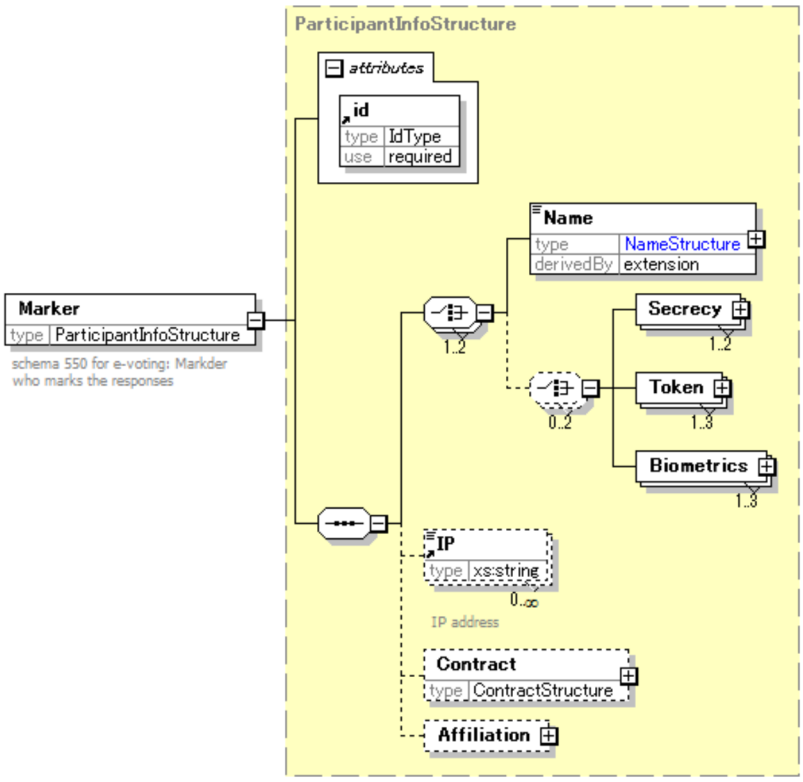

Figure A.94: Marker diagram.

Fig. A.94 shows the schema about **Marker**. This schema is used to define the marker information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.4.63 610-Auditing

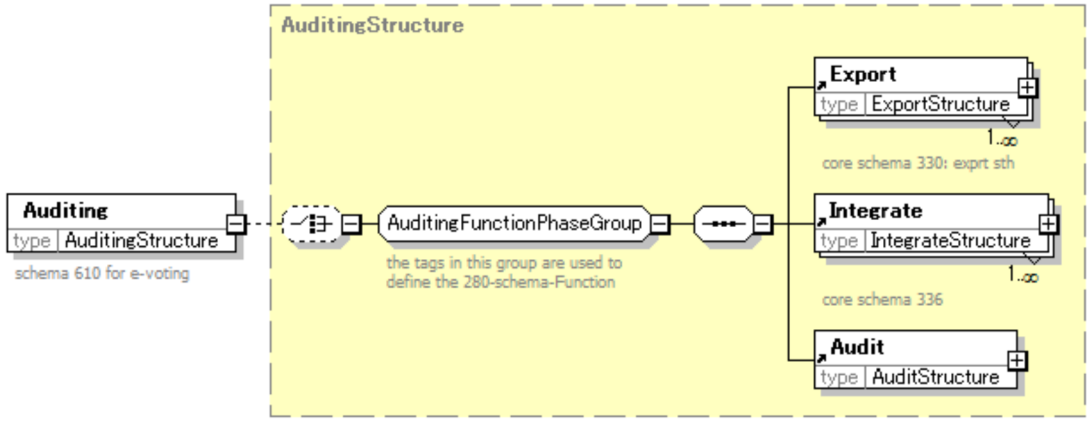

Figure A.95: Auditing diagram.

Element	Group	Remark
Auditing	iAuditingFunctionPhaseGroup	belong to 280 schema

Fig. A.95 shows the schema about **Auditing**. This schema is used to specify the phase for audit the result with respondent numbers and responses in e-voting. This schema is based on the complex type **AuditingStructure**. Because all the phases are listed in **Function** and **Setting**, We are aimed to define each phase has two groups one for function definition, another for setting definition. But until now we do not define the settings during this phase. The elements in the function group are explained in core schemas (schema 330 and 336), please refer to the corresponding schema definitions. As the last child element, **Audit** is not listed in the schema file, because it has the duplicated meanings with other tags.

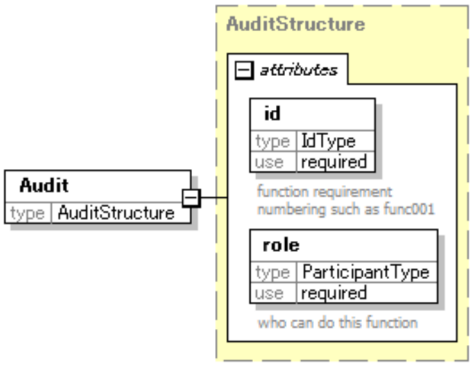

Figure A.96: Audit diagram.

Element	Attribute	Type	Use
Audit	id	IdType	required
	role	ParticipantType	required

Fig. A.96 shows the schema about **Audit**. This schema is used to define the audit function. The system should provide interface to audit result is eligible or not.

A.4.64 620-Candidate

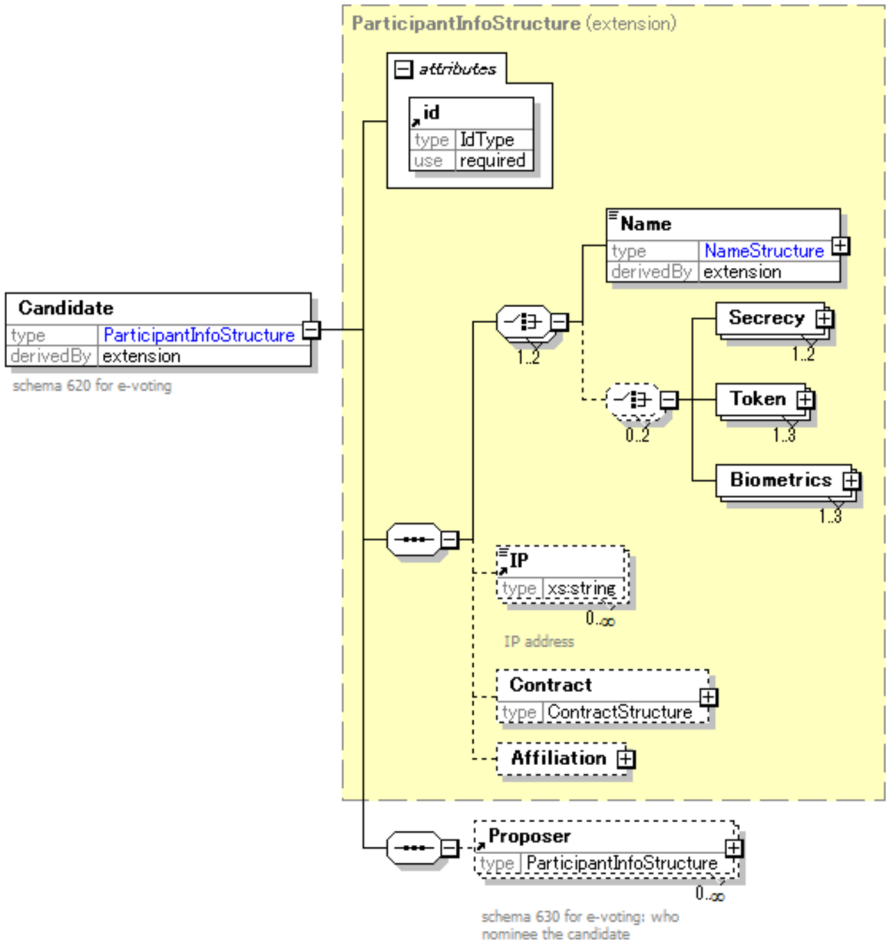

Figure A.97: Candidate diagram.

Fig. A.97 shows the schema about **Candidate**. This schema is used to define the candidate information. The schema is based on the complex type of **ParticipantInfoStructure**. In addition, the element has a (the last) child element named Proposer (schema 630).

A.4.65 630-Proposer

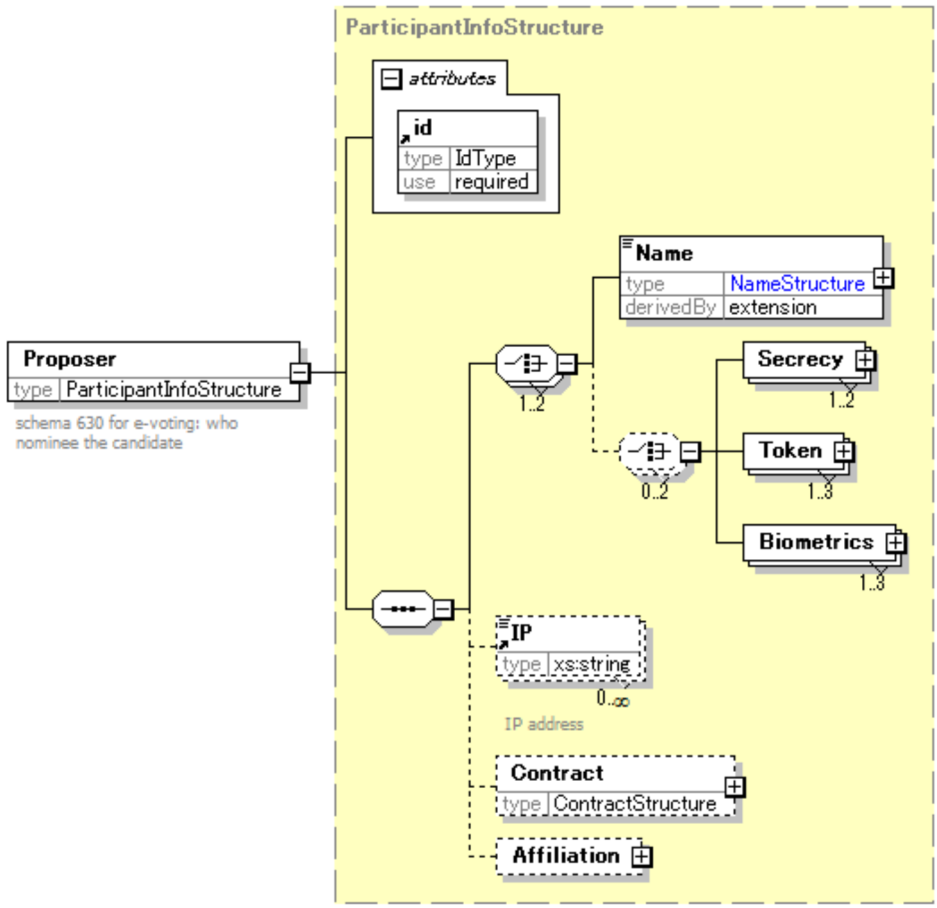

Figure A.98: Proposer diagram.

Fig. A.98 shows the schema about **Proposer**. This schema is used to define the proposer information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.4.66 640-Auditor

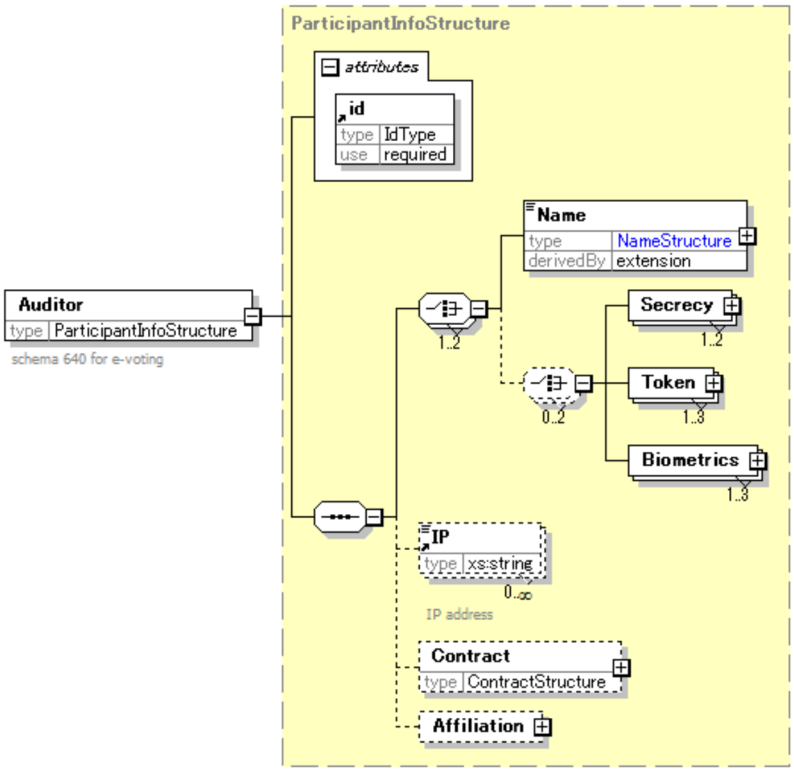

Figure A.99: Auditor diagram.

Fig. A.99 shows the schema about **Auditor**. This schema is used to define the auditor information. The schema is based on the complex type of **ParticipantInfoStructure**.

A.5 Question Types Reference Guide

The whole things of QSL: QSL structure, QSL Schema, elements, attributes, simple types, and complex types are listed previously. In this section, we will see how we can use the important but difficult parts of QSL usage: the question types which determine a complete e-questionnaire, and an e-testing. The reason is that an e-questionnaire or an e-testing is mainly composed of some questions.

Although many e-questionnaire and e-testing systems provide a wide variety of question types particularly in the various classification methods, we concluded four basic types and aimed at gaining various question types through the combination of basic types.

We generally talked about the type attribute of question element in the previous section. Not enough is known about type attribute to use it well. Therefore, we will deeply illustrate the question types and how to use them.

This section provides a reference to all of the question types as follow:

- Basic question types;
- Possible combinations.

All content and diagrams in this section refer to **QuestionPro**.

A.5.1 Basic Question Types

There are four basic question types are listed as following:

- Multiple Choice;
- Open-ended Text;
- Matrix;
- Ranking.

1. **Multiple Choice**

   What is multiple choice question type?

   Multiple Choice is tacit recognition as a radio button or a checkbox that allows the respondents to choose only one or multiple selections from a predefined set of options. In general, there are two types of multiple choice question type, which are the single choice and multiple choices. Some Screenshot help deeply understand it.

   How to distinguish a single choice and multiple choices?

   With Designing QSL based on the combination of primitive elements, both single choice and multiple choices are the multiple choices question type (Drop-down list belongs to single choice). We distinguish both of them by **Limitation**.

   How to set up multiple choice question type?

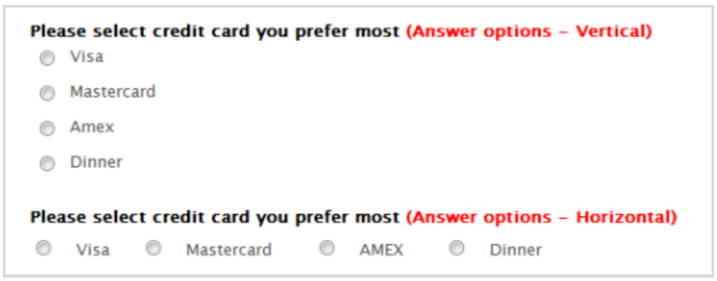

Figure A.100: Screenshot of radio button single choice.

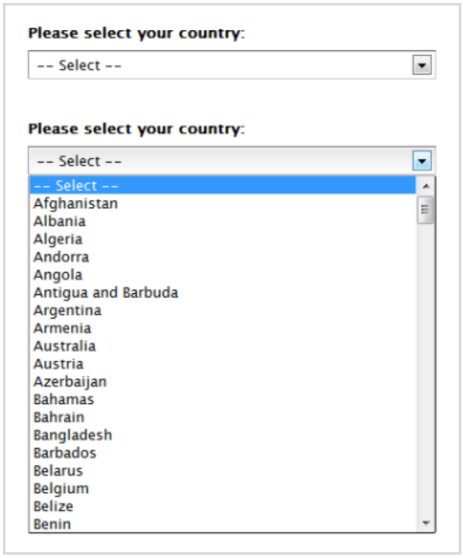

Figure A.101: Screenshot of drop-down single choice.

Step1: to choose question type for **Question**, which has an attribute type and to choose its value "selection."

Step 2: to distinguish the single choice and multiple choices, and to control it by **Limitation**.

- **Tips 1**: As to a radio button single choice type, the questioner should choose the **Limitation** *minOccur* attribute value as "1," *macOccur* attribute value as "1," which means to restrict respondent to select only one answer.
- **Tips 2**: As to a checkbox multiple choices type, the questioner should choose the **Limitation** *minOccur* attribute value more than 1, and *macOccur* attribute value less than the total number of answers, which means to restrict respondent to select multiple answers more than one less than the total number of answers.

Step 3: to set up **Answer**, which has its child element **Description**, which

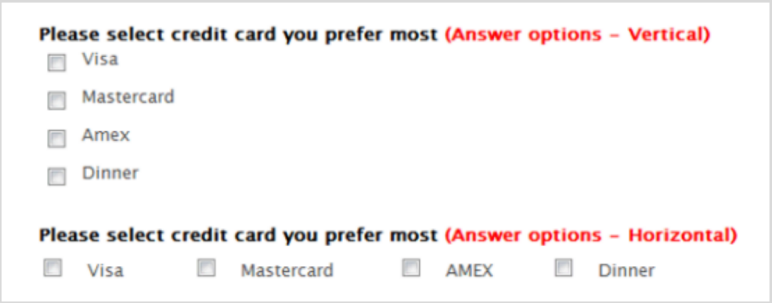

Figure A.102: Screenshot of checkbox multiple choices.

should be written the content of each answer between start-tag and end-tag of text.

As an example to let you deeply understand how to set up a multiple choice question type (radio button single choice):

```
1 <Question isMandatory="yes" type="selection" id="q0001">
2         <Limitation minOccur="1" maxOccur="1"></Limitation
              >
3         <Description type="paragraph" value="Please
              selected credit card u prefer most" id="de0001
              "/>
4         <Answer id="a0001">
5                 <Description type="paragraph" value="Visa"
                      id="de0002"/>
6         </Answer>
7         <Answer id="a0002">
8                 <Description type="paragraph" value="Visa"
                      id="de0003"/>
9         </Answer>
10        <Answer id="a0003">
11                <Description type="paragraph" value="Visa"
                      id="de0004"/>
12        </Answer>
13        <Answer id="a0004">
14                <Description type="paragraph" value="Visa"
                      id="de0005"/>
15        </Answer>
16 </Question>
```

To distinguish the arrangement of Fig. A.100 and Fig. A.101, we can set up **Alignment** element.

```
1 <Question isMandatory="yes" type="selection" id="q0001">
2         <Limitation minOccur="1" maxOccur="1"></Limitation
              >
3         <Description type="paragraph" value="Please
              selected credit card u prefer most" id="de0001
              "/>
4         <Alignment vertical="center"/>
```

```
5           <Answer id="a0001">...
6                   <Alignment dropDown="listBox"/>
7           </Answer>
8 </Question>
```

As another example to let you deeply understand how to set up a multiple choice question type (checkbox multiple choices):

```
1 <Limitation minOccur="1" maxOccur="4"></Limitation>
2 <!--at least choose 1 option and at most choose 4 options
      -->
3 <Limitation minOccur="3" maxOccur="3"></Limitation>
4 <!--Exactly choose 3 options-->
```

To distinguish the arrangement of Fig. A.102, we can set up **Alignment**.

Attention: there is no need to set up a drop-down list for a checkbox selection.

```
1 <Alignment horizontal="left"/>
```

There is a special question type named "True-False." The template snippet is shown below.

```
1 <Question isMandatory="yes" type="selection" id="q0001">
2           <Limitation minOccur="1" maxOccur="1"></Limitation
              >
3           <Description type="paragraph" value="True-False"
              id="de0001"/>
4           <Answer id="a0001">
5                   <Description type="paragraph" value="true"
                      id="de0002"/>
6           </Answer>
7           <Answer id="a0002" isDefault="yes">
8           <!--sample answer-->
9                   <Description type="paragraph" value="false
                      " id="de0003"/>
10          </Answer>
11          <Answer id="a0002" isOtherField="a0003" score="
              -0.5">
12          <!--allow respondent to add another answer,
              usually in e-questionnaire -->
13          <!--if respondent adds other answer, then minus
              0.5 score-->
14                  <Description type="paragraph" value="other
                      " id="de0004"/>
15          </Answer>
16 </Question>
```

2. **Open-ended Text**

   What is open-ended text question type?

   Open-ended Text is a text type question where respondents can input long text, single row text, numeric text, or an email address.

How to distinguish long text, single row text, numeric text, email address?

Because Design of QSL is based on combination of entities, so we proposed that all of these types are the Open-ended Text Question Type. To help understand deeply, Fig. A.103, Fig. A.104, Fig. A.105, and Fig. A.106 are presented below.

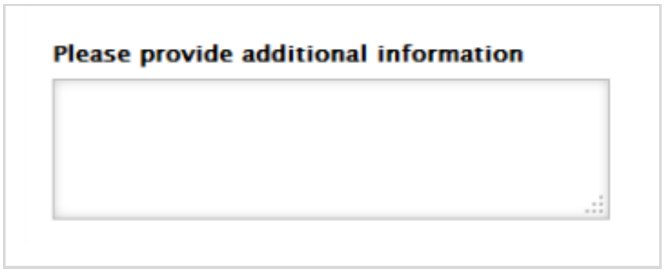

Figure A.103: Screenshot of long text.

Figure A.104: Screenshot of single row text.

Figure A.105: Screenshot of numeric text.

Questioners can use **Limitation** element to restrict.

How to set up open-ended text question type?

At first, questioner should choose the question type for the **Question** element, which has an attribute *type*. Questioner should choose the type attribute value "text."

Attention: If the character text is presented up above the input box, please write in the **Description** element of **Question** element. If the character text is presented in placeholder of the input box, please write in the **Description** element of **Answer** element.

In addition, to set up the **Answer** element, **Answer** element has its child element **Description**.

At last, to distinguish the long text, single row text, numeric text, and email address, questioner can control it through **Limitation** element. However, it

Figure A.106: Screenshot of email address.

is different with Multiple Choice Question Type, the **Limitation** element is as a child element of **Answer** element.

- As to a long text type (developer should set up the default size as 80px × 320px), the questioner should choose the limitation attribute *minLength* value and *maxLength* value less than "255" as default.
- As to a single row text type (developer should set up the default size as 400px, but height cannot be increased), the questioner should choose the **Limitation** *minLength* value and *maxLength* value less than "255" as default.
- As to a numeric text type (developer should set up the default size as 80px, but height cannot be increased), questioner should choose the **Limitation** *minLength* value and *maxLength* value less than "255" as default.
- As to an email address type (developer should set up the default size as 320px, but height cannot be increased), questioner should choose the **Limitation** attribute. In common, the most attribute value and the minLength attribute are not necessary.
- There are 2 **Limitation** locations, first, as the **Question**'s child element, second, as the Answer's child element. The first one in the outer layer to control line numbers. The second one in the inner layer to control character numbers, and the format value to control content format.

As an example to let you deeply understand how to set up a long text type:

```
1 <Question isMandatory="yes" type="text" id="q0001">
2         <Limitation minLength="0" maxLength="5"/>
3         <!--outer layer: limitation control line numbers
              -->
4         <Description type="paragraph" value="Please
              provide additional information" id="de0001"/>
5         <Answer id="a0001">
6                 <Limitation minLength="5" maxLength="255"
                      format="long text"/>
7                 <Description type="paragraph" value="" id=
                      "de0002"/>
8                 <!--inner layer: limitation control
                      character numbers -->
9         </Answer>
10 </Question>
```

As an example to let you deeply understand how to set up a single row text type:

```
<Question isMandatory="yes" type="text" id="q0001">
        <Limitation minLength="1" maxLength="1"/>
        <!--outer layer: limitation control line numbers
            -->
        <Description type="paragraph" value="Name:" id="
            de0001"/>
        <Answer id="a0001">
                <Limitation minLength="2" maxLength="129"
                    format="char"//>
                <Description type="paragraph" value="" id=
                    "de0002"/>
                <!--inner layer: limitation control
                    character numbers -->
        </Answer>
</Question>
```

As an example to let you deeply understand how to set up a numeric input type:

```
<Question isMandatory="yes" type="text" id="q0001">
        <Limitation minLength="1" maxLength="1"/>
        <!--outer layer: limitation control line numbers
            -->
        <Description type="paragraph" value="Phone Number:
            " id="de0001"/>
        <Answer id="a0001">
                <Limitation minLength="5" maxLength="25"
                    format="int"//>
                <Description type="paragraph" value="" id=
                    "de0002"/>
                <!--inner layer: limitation control
                    character numbers -->
        </Answer>
</Question>
```

As an example to let you deeply understand how to set up an email address type:

```
<Question isMandatory="yes" type="text" id="q0001">
        <Limitation minLength="1" maxLength="1"/>
        <!--outer layer: limitation control line numbers
            -->
        <Description type="paragraph" value="Email:" id="
            de0001"/>
        <Answer id="a0001">
                <Limitation minLength="10" maxLength="129"
                     format="char"//>
                <Description type="paragraph" value="" id=
                    "de0002"/>
                <!--inner layer: limitation control
                    character numbers -->
        </Answer>
```

```
10 </Question>
```

There is a special question type named "fill in the blank". The usual presentation in HTML is:

```
1 <p>my name is<input type="text">, I am a student</p>
```

The template snippet is shown below. It concerns the piping logic type and its grammar.

```
1 <Question isMandatory="yes" type="text" id="q0001">
2         <Limitation minLength="1" maxLength="1"/>
3         <!--outer layer: limitation control line numbers
              -->
4         <Description type="paragraph" value="My name is
              $('a0001'), I am a student" id="de0001"/>
5         <Answer id="a0001">
6                 <Limitation minLength="2" maxLength="5"
                      format="char"//>
7                 <Description type="paragraph" value="" id=
                      "de0002"/>
8                 <!--inner layer: limitation control
                      character numbers -->
9         </Answer>
10 </Question>
```

3. **Matrix**

   What is matrix question type?

   Matrix Question Type is a series of questions that share the same answer choices. There are two types, which are one selection type and many selections type.

   How to distinguish one selection type and many selections type?

   One Selection Type: This question can be used when we need to group questions that have the same answer option scale. Consolidated reporting can be done on Matrix questions (see Fig. A.107).

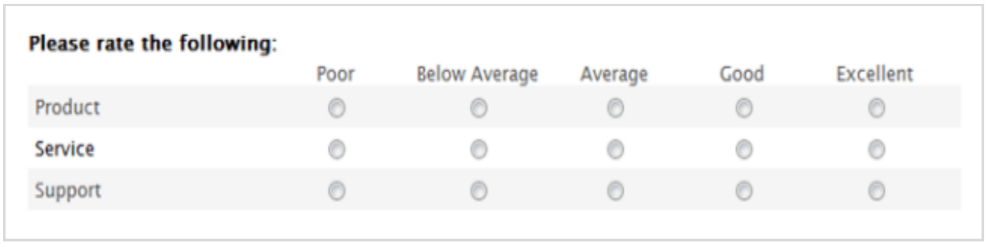

Figure A.107: Screenshot of one selection type.

Many Selection Type: Matrix question where respondents can select multiple options (see Fig. A.108).

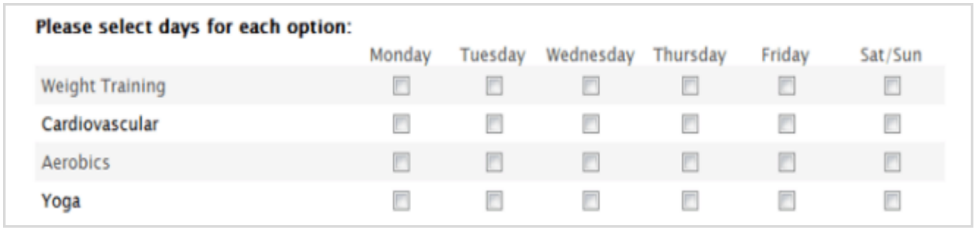

Figure A.108: Screenshot of many selections type.

How to set up a matrix question type?

At first, questioner should choose the question type for the **Question** element, which has an attribute "type." The questioner should choose the type attribute value "matrix."

In addition, to distinguish the one selection and many selections, questioner can control it through **Limitation** element.

- As to a one selection type, questioner should choose the **Limitation** *maxOccur* attribute value as "1," *minOccur* attribute value as "1," which means to restrict respondent to select only one option in each row.
- As to a checkbox multiple choices type, questioner should define the **Limitation** *maxOccur* attribute value not less than 1, *minOccur* attribute value less than total number of answers, which means to restrict respondent to select multiple answers more than one less than total number of options in each row.

At last, to set up the **Answer** element, Answer element has its child element **Description**, which should be written the *value*. In addition, its attributes *Column.Grid* and *Row.Grid* to control the descriptions of column and row.

As an example to let you deeply understand how to set up one selection type:

```
1 <Question isMandatory="yes" type="matrix" id="q0001">
2         <Limitation minOccur="1" maxOccur="1" />
3         <Description type="paragraph" value="Please rate
              the following" id="de0001" />
4         <!--Column-->
5         <Answer id="a0001" Column.Grid="0">
6         <!--Grid begins from 0 -->
7                 <Description type="paragraph" value="Poor"
                      id="de0002" />
8         </Answer>
9         <Answer id="a0002" Column.Grid="1">
10                <Description type="paragraph" value="Below
                      Average" id="de0003" />
11        </Answer>
12         ...
13        <!--Rows-->
```

```
14          <Answer id="a0003" Row.Grid="0">
15                  <Description type="paragraph" value="
                        Product" id="de0004"/>
16          </Answer>
17          <Answer id="a0004" Row.Grid="1">
18                  <Description type="paragraph" value="
                        Service" id="de0005"/>
19          </Answer>
20          <Answer id="a0005" Row.Grid="2">
21                  <Description type="paragraph" value="
                        Support" id="de0006"/>
22          </Answer>
23 </Question>
```

As an example to let you deeply understand how to set up many selections type, it only needs to change the **Limitation**'s attributes.

There is a special example that is a Not Applicable Option in Fig. A.109.

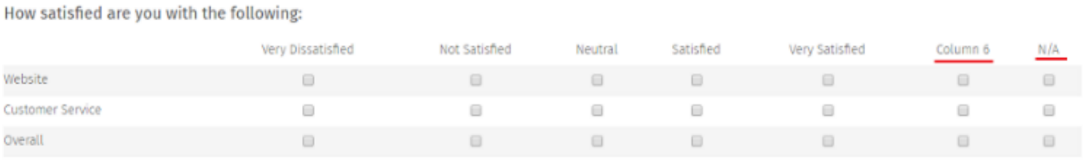

Figure A.109: Screenshot of N/A option in matrix.

```
1 <Answer id="a0002" Column.Grid="6" isNAOption="yes">
2 <!--set NA Option-->
3          <Description type="paragraph" value=" " id="de0007
               "/>
4          <!--omit the value, automatically populate-->
5 </Answer>
```

4. **Ranking**

   What is ranking question type?

   Ranking Question Type allows a certain set of brands or products to be ranked based upon a specific attribute or characteristic. In common, some e-questionnaire systems present rank order type (see Fig. A.110), which is similar with numeric text type to enter the order number.

   How to set up a ranking question type?

   At first, questioner should choose the question type for the **Question** element, which has an attribute *type*. Questioner should choose the type attribute value "ranking."

   Attention: If the value is selected as "ranking" that means the order number cannot be repeated and from 1 to the number of total options. In addition, questioner can use Limitation element to restrict *maxOccur* and *minOccur*.

Figure A.110: Screenshot of ranking question type.

Furthermore, to set up the **Answer** element, answer element has its child element **Text**, which should be written the content of each answer between start-tag and end-tag of text.

At last, do not forget to give it a restriction for its **Alignment** and set up *vertical*.

As an example to let you deeply understand how to set up one selection type:

```
<Question isMandatory="yes" type="ranking" id="q0001">
        <Limitation minLength="1" maxLength="5"/>
        <Description type="paragraph" value="Please rank
            the follow actors:" id="de0001"/>
        <Alignment vertical="stretch"/>
        <Answer id="a0001">
                <Description type="paragraph" value="
                    Johnny Depp" id="de0002"/>
        </Answer>
        <Answer id="a0002">
                <Description type="paragraph" value="Will
                    Smith" id="de0003"/>
        </Answer>
        <Answer id="a0003">
                <Description type="paragraph" value="
                    Leonardo DiCaprio" id="de0004"/>
        </Answer>
        <Answer id="a0004">
                <Description type="paragraph" value="Brad
                    Pitt" id="de0005"/>
        </Answer>
        <Answer id="a0005">
                <Description type="paragraph" value="
                    George Clooney" id="de0006"/>
        </Answer>
</Question>
```

However, there is a difficult that is how to set the sample answer for the e-testing. Even we always set the sample in a Boolean type, but in ranking

situation, the sample should be an order result. Therefore,

```
1 <Question isMandatory="yes" type="ranking" id="q0001">
2          ...
3          <Answer id="a0005" defaultValue="4">
4                  <Description type="paragraph" value="
                       George Clooney" id="de0006"/>
5          </Answer>
6 </Question>
```

A.5.2 Possible Combinations of Question Types

After we explained the basic question types, we will present the possible combinations of basic question types. In this sub-chapter, we will describe the possible combinations as such examples.

At first, we explain some common possible combination question types.

1. **Contact Information**

   What is Contact Information?

   Contact Information adds in a properly formatted and consolidated question asking users for their contact information (see Fig. A.111). It is similar with open-ended text question type, like a set of text questions.

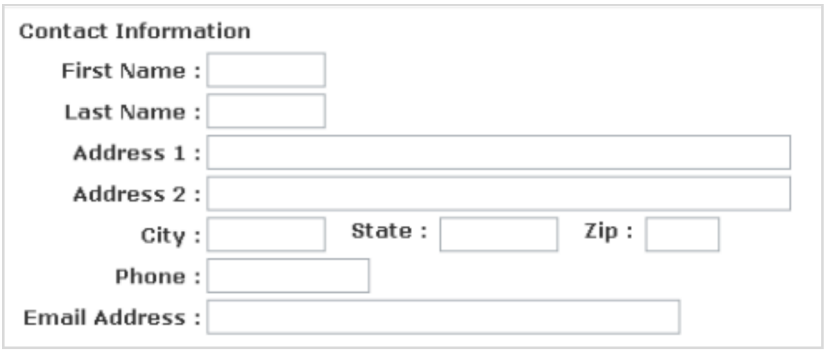

Figure A.111: Screenshot of contact information.

   How to set up a contact information question type?

   At first, questioner should choose the question type for the **Question** element, which has an attribute *type*. Questioner should choose the type attribute value "text," because it likes a set of open-ended text questions.

   In addition, to set up the **Answer** element.

   Furthermore, to distinguish the long text, single row text, numeric text and email address, questioner can control it through **Limitation** element. However, it is different with Multiple Choice Question Type, the **Limitation** element is as a child element of **Answer** element.

At last, the **Alignment** element is necessary to distinguish each line and arrangement.

As an example to set up a contact information presented in Fig. A.111,

```
<Question isMandatory="yes" type="text" id="q0001">
        <Limitation minLength="1" maxLength="1" />
        <Description type="paragraph" value="Contact
            Information" id="de0001"/>
        <Alignment vertical="stretch"/>
        <!--each answer arrangement-->
        <Answer id="a0001">
                <Limitation minLength="1" maxLength="15"
                    format="char"/>
                <Description type="paragraph" value="First
                     Name:" id="de0002"/>
        </Answer>
        <Answer id="a0002">
                <Limitation minLength="1" maxLength="15"
                    format="char"/>
                <Description type="paragraph" value="First
                     Name:" id="de0003"/>
        </Answer>
        <Answer id="a0003">
                <Limitation minLength="5" maxLength="129"
                    format="char"/>
                <Description type="paragraph" value="
                    Address 1:" id="de0004"/>
        </Answer>
        <Answer id="a0004">
                <Limitation minLength="5" maxLength="129"
                    format="char"/>
                <Description type="paragraph" value="
                    Address 2:" id="de0005"/>
        </Answer>
        ...
</Question>
```

2. **Matrix Spreadsheet**

   What is matrix spreadsheet?

   Matrix Spreadsheet is a multiple questions in spreadsheet style with text input boxes (see Fig. A.112). It is the combination of open-ended text type and matrix. In QSL, we sort it as one of the matrix (easy to remember because of the name of it).

```
<Question isMandatory="yes" type="matrix" id="q0001">
        <Limitation minLength="1" maxLength="5" format="
            int"/>
        <!--length control the character -->
        <Description type="paragraph" value="Please rate
            the following" id="de0001"/>
        <!--Column-->
        <Answer id="a0001" Column.Grid="0">
```

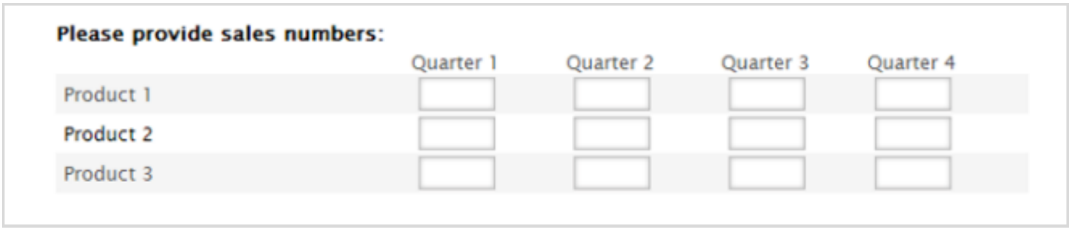

Figure A.112: Screenshot of matrix spreadsheet.

```
 7                   <!--Grid begins from 0 -->
 8                   <Description type="paragraph" value="
                         Quarter 1" id="de0002"/>
 9           </Answer>
10           <Answer id="a0002" Column.Grid="1">
11                   <Description type="paragraph" value="
                         Quarter 2" id="de0003"/>
12           </Answer>
13           <Answer id="a0003" Column.Grid="2">
14                   <Description type="paragraph" value="
                         Quarter 3" id="de0004"/>
15           </Answer>
16           <Answer id="a0004" Column.Grid="3">
17                   <Description type="paragraph" value="
                         Quarter 4" id="de0005"/>
18           </Answer>
19           <!--Rows-->
20           <Answer id="a0005" Row.Grid="0">
21                   <Description type="paragraph" value="
                         Product 1" id="de0006"/>
22           </Answer>
23           <Answer id="a0006" Row.Grid="1">
24                   <Description type="paragraph" value="
                         Product 2" id="de0007"/>
25           </Answer>
26           <Answer id="a0007" Row.Grid="2">
27                   <Description type="paragraph" value="
                         Product 3" id="de0008"/>
28           </Answer>
29 </Question>
```

3. **Slide and Rating**

   These 4 kinds of question types (Fig. A.113, Fig. A.114, Fig. A.115, and Fig. A.116) are similar. From the respective of basic question type, we consider them as single selection. They have different presentations. However, QSL does not consider the front-end presentation.

   Star Rating can be designed as a single selection question with 5 answer options, which are weights from 1to 5. Smiley Rating can be designed as a single selection question with 5 answer options and their corresponding images. As to Slide question, it also can be designed as a single selection

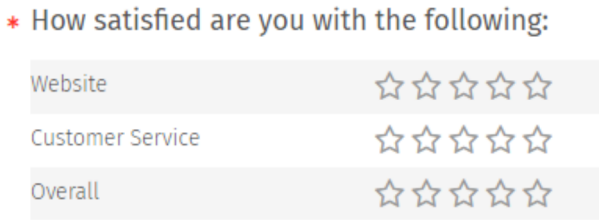

Figure A.113: Screenshot of star rating.

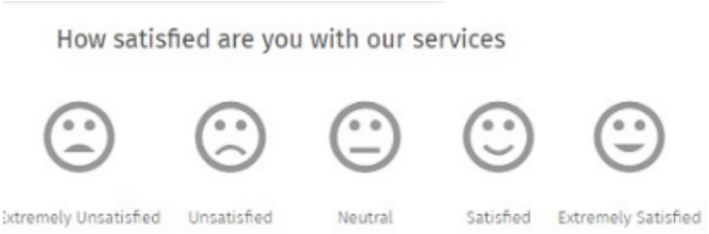

Figure A.114: Screenshot of smliey rating.

with a 100 options. In this situation, questioner just need to specify the limitation *maxOccur* as 100, and omits the answer values and ids.

4. **Rank Order and Drag**

   This is based on the single selection (drop-down list) in Fig. A.117, but concerns exclusive when choose an option, the others cannot choose the selected options. As the prototype, It likes a matrix: grid row (skiing, snowboarding, biking), and grid column (1, 2, 3), and mutually exclude.

   This is based on ranking question type in Fig. . Drag is a presentation, which QSL does not concern until now.

```
1 <Question isMandatory="yes" type="ranking" id="q0001">
2 <!--question type: ranking-->
3         <Description type="paragraph" value="Please rank
              ..." id="de0001"/>
```

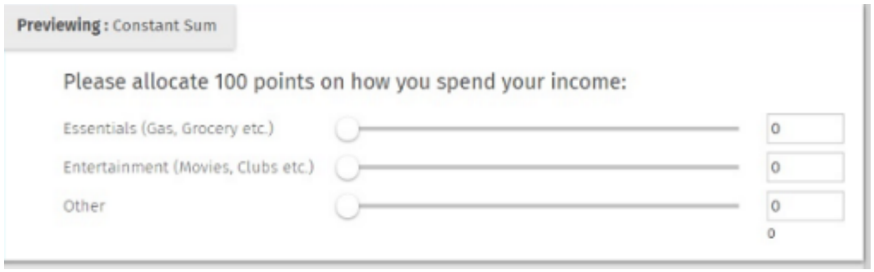

Figure A.115: Screenshot of slide.

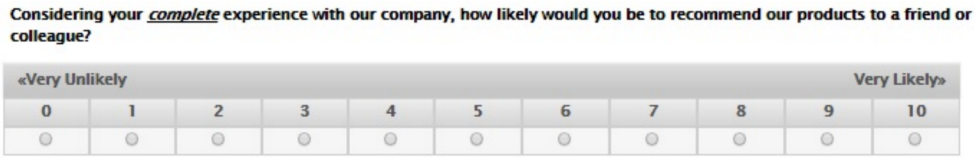

Figure A.116: Screenshot of NPS.

Figure A.117: Screenshot of rank order.

```
4         <Answer id="a0001">
5                 <Description type="paragraph" value="
                      Skiing" id="de0002"/>
6         </Answer>
7         <Answer id="a0002">
8                 <Description type="paragraph" value="
                      Snowboarding" id="de0003"/>
9         </Answer>
10        <Answer id="a0003">
11                <Description type="paragraph" value="
                      Biking" id="de0004"/>
12        </Answer>
13 </Question>
```

This image refers to QTI in Fig. A.119. It is based on ranking (unrepeated options), but this question concerns the image and location. The questioner can mark the locations on the map and let respondent to choose. It is an extension of "match item" or "Connect the Point" question type.

```
1 <Question isMandatory="yes" type="ranking" id="q0001">
2         <Limitation minOccur="5" maxOccur="5"/>
3         <Description type="header" value="AIRPOER TAGS" id
              ="de0001"/>
4         <!--header -->
5         <Description type="paragraph" value="The
              International Air Transport..." id="de0002"/>
6         <!--explanation-->
7         <Media id="me0001" type="img/png" src="img/airport
              .png"/>
8         <!--Image with marks -->
9         <Answer id="a0001">
```

Figure A.118: Screenshot of drag object.

```
10                  <Description type="paragraph" value="EGB"
                        id="de0003" />
11          </Answer>
12          ...
13 </Question>
```

There is another similar type in Fig. A.120.

5. **Upload File**

```
1 <Question isMandatory="yes" type="upload" id="q0001">
2 <!--question type: upload-->
3          <Description type="paragraph" value="Drag your
               file here" id="de0001" />
4          <Answer id="a0001">
5                  <Description type="paragraph" value=" " id
                       ="de0002" />
6                  <!--omit value -->
7          </Answer>
8 </Question>
```

6. **Sid-by-side Matrix**

This is a 3D-matrix question type. X.Grid (Importance, Satisfaction), Y.Grid (1,2,3,4,5), Z.Grid (Customer Service, Product Package, On-Time Arrivals) in Fig. A.122.

```
1 <Question isMandatory="yes" type="matrix" id="q0001">
2          <Limitation minOccur="1" maxOccur="1" /><!--single
               selection-->
3          <Description type="paragraph" value="Please rate
               the following" id="de0001" />
4          <!--X-->
5          <Answer id="a0001" X.Grid="0">
6                  <Description type="paragraph" value="
                       Importance" id="de0002" />
```

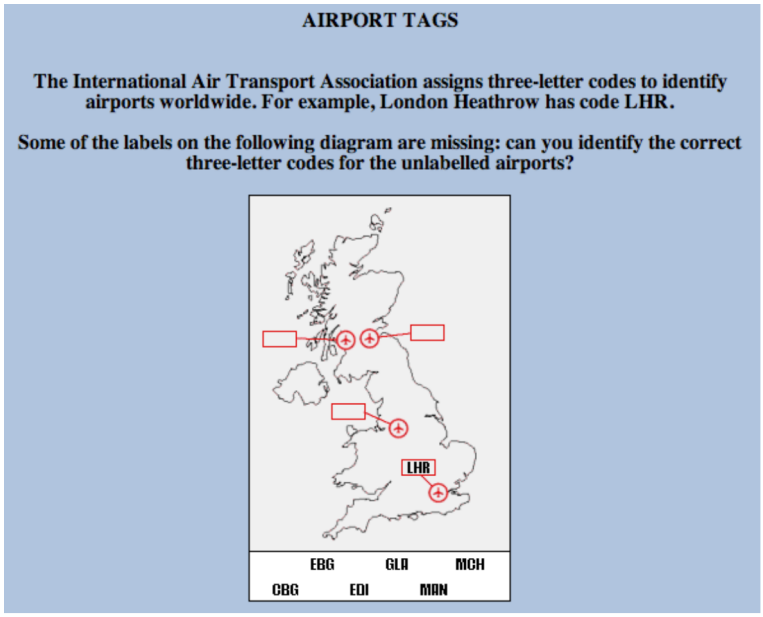

Figure A.119: Screenshot of image of drag target.

```
7           </Answer>
8           <Answer id="a0002" X.Grid="1">
9                   <Description type="paragraph" value="
                        Satisfaction" id="de0003"/>
10          </Answer>
11          <!--Y-->
12          <Answer id="a0003" Y.Grid="0">
13                  <Description type="paragraph" value="1" id
                        ="de0004"/>
14          </Answer>...
15          <!--Z-->
16          <Answer id="a0004" Z.Grid="0">
17                  <Description type="paragraph" value="
                        Customer Service" id="de0005"/>
18          </Answer>...
19 </Question>
```

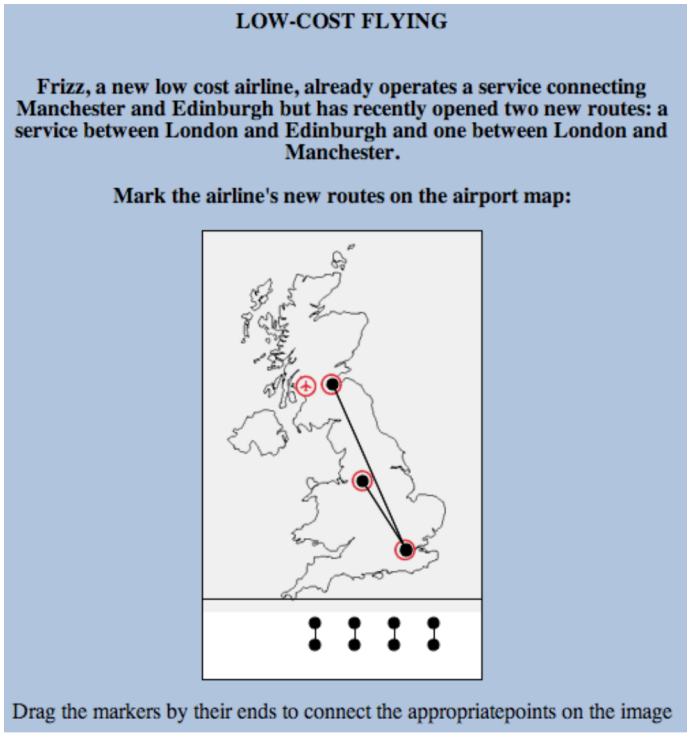

Figure A.120: Screenshot of image of connect point.

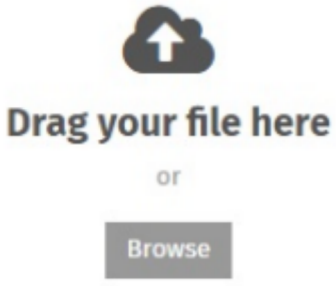

Figure A.121: Screenshot of upload file.

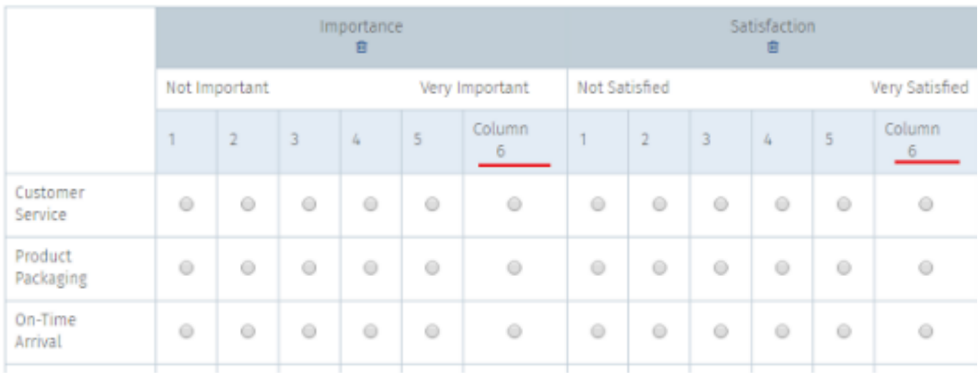

Figure A.122: Screenshot of SBS-matrix.

A.6 Logic Types Reference Guide

So far, we have introduced the creation of static e-questionnaires. In this section, we will see how to create dynamic e-questionnaires; we also called "intelligent" e-questionnaires to ensure that only relevant questions are displayed to the appropriate respondents.

Previously, we described the logic element and its attribute type. However, only scratching the surface, for this large field, was not enough. For reason, it owes to various manifestations of e-questionnaires.

From all the minor details, many e-questionnaire systems provide it a variety of classifications and definitions. And thereupon we proceeded with the thoughts of trichromatic combination principle, in other words, a multiplicity of separate elements according to some standards. As to uses, this section provides a reference to all of logic types (basic logic types and possible combinations) to define them.

All content and diagrams in this section refer to QuestionPro.

A.6.1 Basic Logic Types

There are four basic kinds of logic types:

- Skipping;
- Piping;
- Extraction;
- Randomization.

1. **Skipping**

   What is skipping?

   Skipping gives questioner the ability to create "intelligent" e-questionnaires. For example, it wouldn 't make sense to ask questions about married life to someone who is not married. Using skipping questioner can ensure that only relevant questions are displayed to the appropriate respondents.

   How to set up skipping?

   - The **Logic** is the child of **Paper**, so that questioner must write logic that should be included in tags of paper.
   - **Logic** has a child element **Route**. **Route**'s attribute *type* is written as "skipping".
   - **Route**'s child elements **Condition** and **Action** to control if satisfies conditions and then jump to where.

```
1 <Logic>
2         <Route type="skipping" id="ro0001">
3                 <Condition answerId="a0001" isChecked
                      ="yes" />
```

```
4                   <Action questionId="q0007"/>
5           </Route>
6 </Logic>
```

which means that if selected a0001, then jump to question qu0007.

- Sometimes, between start-tags and the end-tags, records the URL of destination, as to the to attribute, as the sample value as "url."

```
1 <Logic>
2           <Route type="skipping" id="ro0001">
3                   <Condition answerId="a0001" isChecked
                        ="yes"/>
4                   <Action to="www.url.com"/>
5           </Route>
6 </Logic>
```

which means that from this answer skipping to a website. There is a situation that if choose an option then jump to Thank you page usually occurs in e-questionnaire.

What is the different skip to options?

Based on the answer option selected, respondents will be directed to the jump to location that is set on the survey. Following skipping to options are available:

- **Papers**: select from any e-questionnaire and e-testing that is permitted to enter.
- **Questions**: select from any question that is after the source question.
- **Sections**: can select from any of the section that are after the source question.
- **Answers**: usually skip to answer when question types is chosen as "text."
- **Terminate**: go to finished page, if questioner design its setting that respondent can see the result, then show the statistics, otherwise, jump to:
- **Go to Conclusion Page (Description)**: Select this option if questioners wish to direct respondents to the Thank You page. Response will be marked as a complete response. In other words, select this option if you wish to terminate the survey for respondents.
- **Chain URL**: Select this option to take respondents to a different website.

What is Default Destination skipping?

The Default Destination skipping is the fallback logic that gets executed when no other logic gets triggered. If respondents do not choose any option, the default skipping destination is used when NO OTHER logic gets triggered. This is the fallback logic that gets executed when no other logic gets triggered.

```
<Logic>
        <Route type="skipping" id="ro0001">
<Condition questionId="q0001" isChecked="no"/>
                    <!--if there is no choice of question-->
<Action descriptionId="de0010"/>
                    <!--Then finish-->
        </Route>
</Logic>
```

2. **Piping**

   What is piping?

   Piping enables you to carry text from one question to the next depending on the options selected by the respondent. For example if you have a scenario as described below (see Fig. A.123).

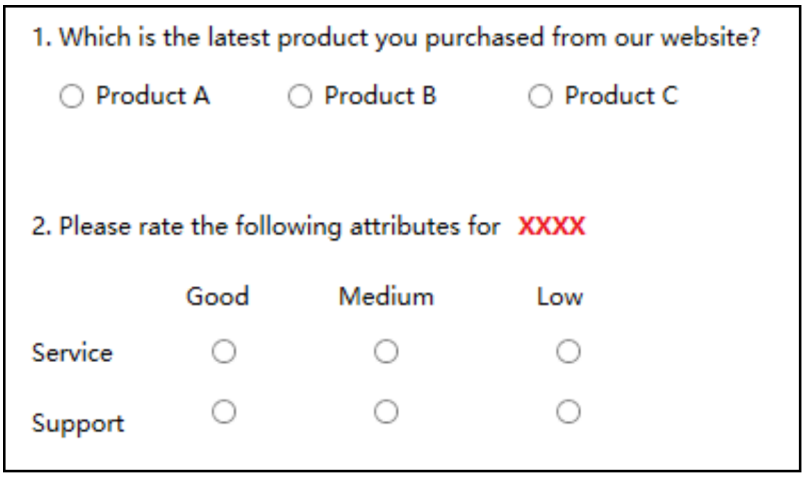

Figure A.123: Screenshot of piping.

Now questioner would like to replace the text XXXX with the actual product that the respondent had selected for question 1. Piping will enable you to replace XXXX with the appropriate text.

How to set up Piping?

- **Logic** is the schema 310 element. Its child element **Route** to define a route of logic. It is necessary to define route id, and type value as "piping."
- **Route**'s child elements: **Condition** and **Action**.
- **Condition** sets the question's reference, and the condition trigger (isChecked). **Action** sets the piping route and the filled location to attribute that is written as dollar mark as braces including piping Ids, underscore and id which is like a function name, such as "$('piping Ids')". We restricted the value of it as "$('pip[0-9]*')." Then, in the destination question. As the example,

```
1 <Question id="q0002">
2         <Description id="de0002" type="paragraph" value="
              Please rate the following attributes for $('
              pip0001')"/>
3 <!--question2 title add the piping location-->
4          ...
5 </Question>
6 ...
7 <Logic>
8         <Route type="piping" id="ro0001">
9                 <Condition paperId="p0001" sectionId="
                      s0001" questionId="q0001" isChecked="
                      yes"/>
10                <Action paperId="p0001" sectionId="s0001"
                      questionId="q0002" isFilled="yes" to="
                      $('pip0001')"/>
11 <!--action: filled the piping location into checked values
       -->
12        </Route>
13 </Logic>
```

In this example, at the first question, there are 3 choices using piping logic, whichever be chosen, the question answer text will be piping to next question as one part of next question text.

Can I carry over the Piping Text to multiple questions?

Piping text can only be carried over to the next question. If you wish to carry over text to multiple questions you can get more information in possible combinations of logic types below.

Is there any limitation on the number of characters that can be piped?

Piping text has a limitation of 255 characters (Including Spaces/Special Characters).

3. **Extraction What is extraction?**

   Extraction enables questioners to display the choices of a question based on the choices selected for the previous question. It differs from piping. Piping is used to carry out the selected options to fill in some descriptions. However, Extraction is used to carry out the selected options to compare the existing question's texts. For example, (see Fig. A.124, Fig. A.125, and Fig. A.126)

   The following question without extraction will be displayed as follows (Fig. A.125):

   The following question without extraction will be displayed as follows (Fig. A.126):

   If the respondent selects options **AOL** and **Earthlink** for question 1 then the extracted question (i.e. question 2) will be displayed as follows:

   (Only the options selected by the respondent will be displayed on e-questionnaire for respondent).

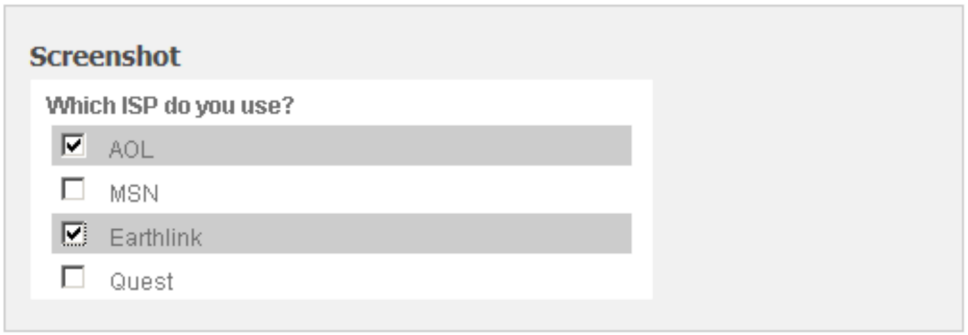

Figure A.124: Screenshot of extraction 1.

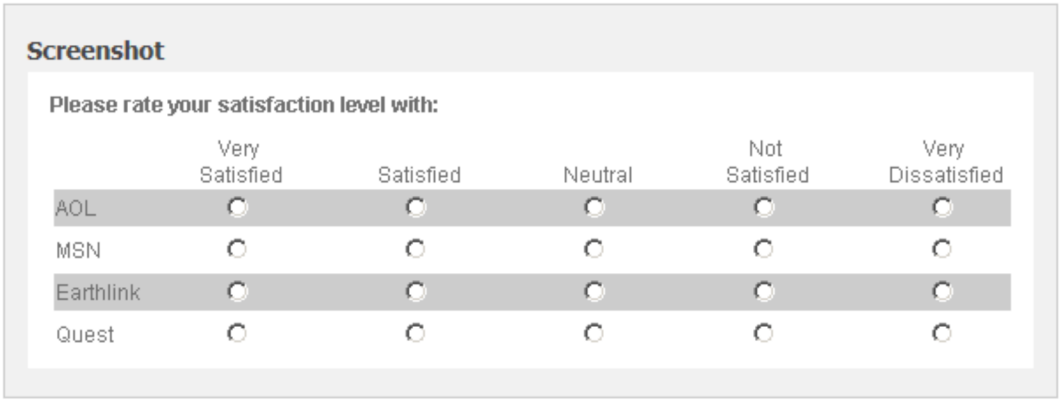

Figure A.125: Screenshot of extraction 2.

How to set up Extraction?

- **Logic** is the schema 310 element. Its child element Route to define a route of logic. It is necessary to define route id, and type value as "extraction."
- **Route**'s child elements: **Condition** and **Action**.
- **Condition** sets the question's reference, and the condition trigger (isChecked). **Action** sets the piping route and the filled location to attribute that is written as dollar mark as braces including piping Ids, underscore and id which is like a function name, such as "$('piping Ids')." We restricted the value of it as "$('ext[0-9]*')." Then, in the destination question.

```
1 <Paper id=" p0001" type="questionnaire">
2         <Section id=" s0001">
3                 <Question isMandatory="yes" type="
                      selection" id="q0001">
4                       <Limitation minLength="1"
                            maxLength="4" />
```

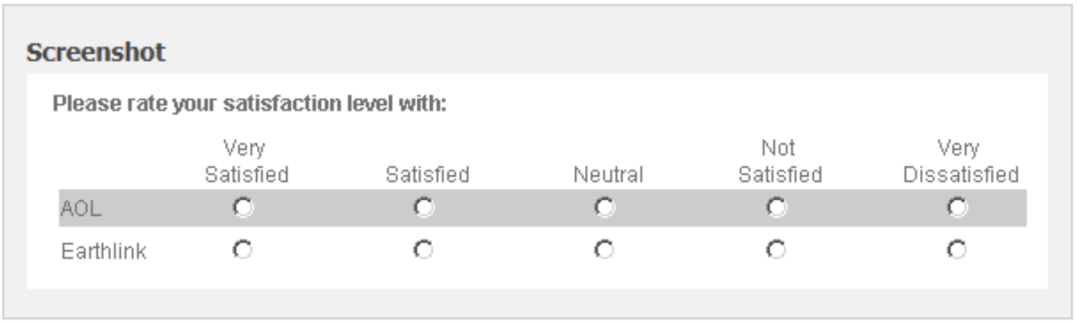

Figure A.126: Screenshot of extraction 3.

```
5                       <Description type="paragraph"
                            value="Which ISP do you use?"
                            id="de0001" />
6                       <Answer id="a0001">
7                               <Description type="
                                    paragraph" value="AOL"
                                     id="de0002" />
8                       </Answer>
9                       <Answer id="a0002">
10                              <Description type="
                                    paragraph" value="MSN"
                                     id="de0003" />
11                      </Answer>
12                      <Answer id="a0003">
13                              <Description type="
                                    paragraph" value="
                                    Earthlink" id="de0004"
                                    />
14                      </Answer>
15                      <Answer id="a0004">
16                              <Description type="
                                    paragraph" value="
                                    Quest" id="de0005" />
17                      </Answer>
18              </Question>
19              <Question isMandatory="yes" type="matrix"
                    id="q0002">
20                      <Limitation minOccur="1" maxOccur=
                            "1" />
21                      <Description type="paragraph"
                            value="Plaese rate your
                            satisfaction level with:" id="
                            de0006" />
22                      <!--Column 0-4-->
23                      <Answer id="a0005" Column.Grid="0"
                            >
24                              <Description type="
                                    paragraph" value="Very
                                     Satisfied" id="de0007
                                    " />
```

```
25                  </Answer>
26                  <Answer id="a0006" Column.Grid="1"
                        >
27                      <Description type="
                            paragraph" value="
                            Satisfied" id="de0008"
                            />
28                  </Answer>
29                  <Answer id="a0007" Column.Grid="2"
                        >
30                      <Description type="
                            paragraph" value="
                            Neutral" id="de0009"/>
31                  </Answer>
32                  <Answer id="a0008" Column.Grid="3"
                        >
33                      <Description type="
                            paragraph" value="Not
                            Satisfied" id="de0010"
                            />
34                  </Answer>
35                  <Answer id="a0009" Column.Grid="4"
                        >
36                      <Description type="
                            paragraph" value="Very
                             Dissatisfied" id="
                            de0011"/>
37                  </Answer>
38                  <!--Normal situation begin: Row
                        0-3-->
39                  <Answer id="a0010" Row.Grid="0">
40                      <Description type="
                            paragraph" value="AOL"
                             id="de0012"/>
41                  </Answer>
42                  <Answer id="a0011" Row.Grid="1">
43                      <Description type="
                            paragraph" value="MSN"
                             id="de0013"/>
44                  </Answer>
45                  <Answer id="a0012" Row.Grid="2">
46                      <Description type="
                            paragraph" value="
                            Earthlink" id="de0014"
                            />
47                  </Answer>
48                  <Answer id="a0013" Row.Grid="3">
49                      <Description type="
                            paragraph" value="
                            Quest" id="de0015"/>
50                  </Answer>
51                  <!--Normal situation end: Row 0-3
                        -->
52                  <!--Extraction situation begin:
                        Rows-->
```

```
53                          <Answer id="a0010" Row.Grid="*"> <
                                !--rows auto-fill |add-->
54                                  <Description type="
                                        paragraph" value="$('
                                        ext0001')" id="de0012"
                                        />
55                          </Answer>
56                          <!--Extraction situation end: Rows
                                -->
57                  </Question>
58          </Section>
59          <Logic>
60                  <Route type="extraction" id="ro0001">
61                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0001" isChecked="yes"/>
62                          <Action paperId="p0001" sectionId=
                                "s0001" questionId="q0002"
                                isFilled="yes" to="$('ext0001
                                ')"/>
63                  </Route>
64          </Logic>
65 </Paper>
```

4. **Randomization**

   What is randomization?

   In order to avoid cheating, some questioners want sections, questions, or answers can be randomized. Randomization enables you to display without ordering in advance.

   How to use randomization?

   Because randomization does not need to piping to a destination, so if the questioner chooses a randomization type as an attribute value of Route. The reference attribute group support the paper randomization, section randomization, question randomization, and answer randomization.

   What is the conflict?

   There is a schema 230 Setting. This schema controls the order setting (ascending, descending, and default). If questioner does not set this setting for the same location (paper, section, question, answer), then he can set this logic route.

   With which question type does the answer display order works?

   - Multiple Choice - Select One, Drop Down, Image Chooser, Select Many, Image Chooser (Select Many)
   - Rank Order / Drag N Drop

   How to set up Randomization?

- **Logic** is the schema 310 element. Its child element **Route** to define a route of logic. It is necessary to define route id, and type value as "randomization."
- **Route**'s child elements: **Condition** and **Action**. In randomization logic situation, it does not need **Condition**.
- **Action** sets the locations should be randomized.

As the example,

```
<Logic>
        <Route type="randomization" id="ro0001">
                <!--The answers of location:( paper 1
                    section 1 question 1) should be
                    randomized -->
                <Action paperId="p0001" sectionId="s0001"
                    questionId="q0001"/>
                <!-- The questions of location:( paper 1
                    section 2) should be randomized ->
                <Action paperId="p0001" sectionId="s0002"/
                    >
                <Action paperId="p0001" sectionId="s0001"
                    questionId="q0001"/>
        </Route>
        <Route type="randomization" id="ro0002">
                <!-- The sections of location:( paper 2)
                    should be randomized -->
                <Action paperId="p0002"/>
        </Route>
</Logic>
```

A.6.2 Possible Combinations of Logic Types

After we explained the basic logic types, we will present the possible combinations of basic logic types. In this sub-chapter, we will describe the possible combinations as such examples.

At first, we explain some common possible combination logic types.

1. **Matrix Extraction**

   What is extraction from matrix question type?

   Extracting from a matrix question is a very useful feature, especially for surveys such as (see Fig. A.127):

   Now, suppose questioner who would like to ask the respondent why he/she selected Not Satisfied for a particular service provider. The questioner can use **Extraction** to set this up.

   If the respondent selects Not Satisfied for **AOL** and **Qwest** for question in Fig. A.127 then the extracted question (i.e. in Fig. A.128) will be displayed as follows:

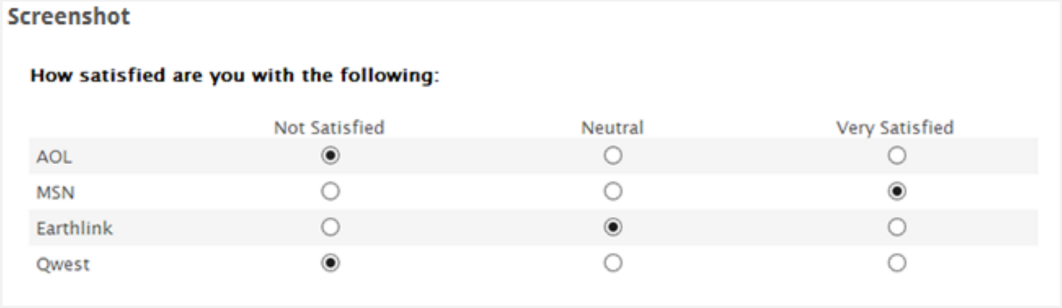

Figure A.127: Screenshot of matrix extraction 1.

Figure A.128: Screenshot of matrix extraction 2.

2. **Compound Branching**

   What is compound branching logic?

   With simple branching of skipping type, questioners cannot set up logic based on responses to multiple questions. Also with simple branching, the logic is executed immediately. **Compound Branching** means that with help of compound Logic you can set up branching based on responses to multiple questions in the questionnaire. Sometimes, there is another name called Delayed Branching that means questioners can set up Branching Logic based on a Question which was asked much earlier in the questionnaire. But, there are no differences between both of them.

   How to use Compound Branching?

   In compound branching, we provide logic type " skipping " to jump to other questions, and attribute to connect with relative answers.

   Consider this example:

   - Q1: Gender
   - Q2: Age
   - Q3: General question to be answered by everybody

- Q4: General question to be answered by everybody
- Q5: General question to be answered by everybody

Now after Q5 respondents should branch to different questions based on their selection in Q1 AND Q2

- Q6: To be answered only by respondents who selected Gender=Male and Age=18 & above
- Q7: To be answered only by respondents who selected Gender=Female and Age=18 & above

Specification for setting up this case:

```
1 <Paper type="questionnaire" id="p0001">
2         <Section id="s0001">
3                 <Question isMandatory="yes" type="
                      selection" id="q0001">
4                 <Limitation minOccur="1" maxOccur="1"/>
5                         <Description type="paragraph"
                              value="Gender" id="de0001"/>
6                         <Answer id="a0001">
7                                 <Description type="
                                      paragraph" value="F"
                                      id="de0002"/>
8                         </Answer>
9                         <Answer id="a0002">
10                                <Description type="
                                      paragraph" value="M"
                                      id="de0003"/>
11                        </Answer>
12                </Question>
13                <Question isMandatory="yes" type="
                      selection" id="q0002">
14                        <Limitation minOccur="1" maxOccur=
                              "1"/>
15                        <Description type="paragraph"
                              value="Age" id="de0004"/>
16                        <Answer id="a0003">
17                                <Description type="
                                      paragraph" value="less
                                       than 18" id="de0005"/
                                      >
18                        </Answer>
19                        <Answer id="a0004">
20                                <Description type="
                                      paragraph" value="
                                      greater than or equal
                                      to 18, less than 60"
                                      id="de0006"/>
21                        </Answer>
22                        <Answer id="a0005">
```

```
23                                  <Description type="
                                        paragraph" value="
                                        greater than or equal
                                        to 60" id="de0007"/>
24                          </Answer>
25                  </Question>
26                  <!--Question 3-5 omitted-->
27                  <Question isMandatory="yes" type="text" id
                        ="q0006">
28                          <Limitation minLength="1"
                                maxLength="1"/>
29                          <Description type="paragraph"
                                value="Reason" id="de0013"/>
30                          <Answer id="a0010">
31                                  <Description type="
                                        paragraph" value="" id
                                        ="de0014"/>
32                          </Answer>
33                  </Question>
34                  <Question isMandatory="yes" type="text" id
                        ="q0007">
35                          <Limitation minLength="1"
                                maxLength="1"/>
36                          <Description type="paragraph"
                                value="Reason" id="de0015"/>
37                          <Answer id="a0011">
38                                  <Description type="
                                        paragraph" value="" id
                                        ="de0016"/>
39                          </Answer>
40                  </Question>
41          </Section>
42          <Logic>
43                  <Route type="skipping" id="ro0001">
44                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0001" answerId="a0002"
                                isChecked="yes" relation="and"
                                />
45                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0002" answerId="a0003"
                                isChecked="yes" relation="or"/
                                >
46                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0002" answerId="a0004"
                                isChecked="yes"/>
47                          <Action paperId="p0001" sectionId=
                                "s0001" questionId="q0006"/>
48                  </Route>
49                  Route type="skipping" id="ro0002">
50                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0001" answerId="a0001"
```

```
                                isChecked="yes" relation="and"
                                />
51                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0002" answerId="a0003"
                                isChecked="yes" relation="or"/
                                >
52                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0002" answerId="a0004"
                                isChecked="yes"/>
53                          <Action paperId="p0001" sectionId=
                                "s0001" questionId="q0006"/>
54                  </Route>
55          </Logic>
56 </Paper>
```

Attention: In theory, we can set up a delayed branching, such as use piping function in question 2 in above example, but, it is difficult to read and compile. Therefore, please try to avoid skipping back.

3. **Looping with Text Piping**

   What is looping with text piping?

   Simply put, considering the following example: Questioners are asking respondents to select credit cards they have and for each credit card, and wish to collect further information using follow up questions (see Fig.).

   If the respondents select the options “Visa” and “ American Express ”, they should only be asked follow up questions for “Visa” and “American Express” cards. They should skip questions for “ Master ” and “Discover” cards.

   To set this up, first set up the entire survey with all questions (see Fig. A.129).

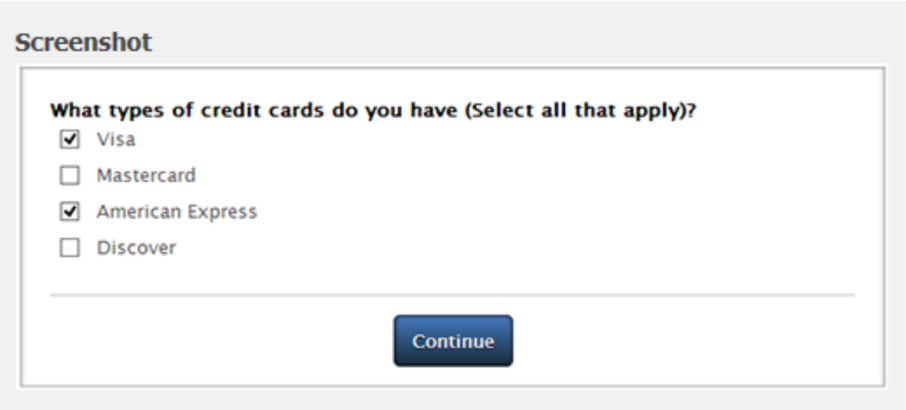

Figure A.129: Screenshot of looping with text piping.

How to use looping with text piping? Looping with Text Piping is derived from Piping. Therefore, the logic type value is “ piping ”. Specification on above example:

```
1 <Paper type="questionnaire" id="p0001" >
2         <Section id="s0001">
3                 <Question isMandatory="yes" type="
                      selection" id="q0001">
4                         <Limitation minOccur="1" maxOccur=
                              "4" />
5                         <Description type="paragraph"
                              value="What types of credit
                              cards do you have (Select all
                              that apply?)" id="de0001" />
6                         <Alignment vertical="stretch" />
7                         <Answer id="a0001">
8                                 <Description type="
                                      paragraph" value="Visa
                                      " id="de0002" />
9                         </Answer>
10                        <Answer id="a0002">
11                                <Description type="
                                      paragraph" value="
                                      Mastercard" id="de0003
                                      " />
12                        </Answer>
13                        <Answer id="a0003">
14                                <Description type="
                                      paragraph" value="
                                      American Express" id="
                                      de0004" />
15                        </Answer>
16                        <Answer id="a0004">
17                                <Description type="
                                      paragraph" value="
                                      Discover" id="de0005" /
                                      >
18                        </Answer>
19                        <Description type="break" value=""
                              id="de0006" />
20                </Question>
21                <Question isMandatory="yes" type="
                      selection" id="q0002">
22                        <Limitation minOccur="1" maxOccur=
                              "1" />
23                        <Description type="paragraph"
                              value="How oftern do you use
                              your $('pip0001')" id="de0006"
                              />
24                        <Answer id="a0005">
25                                <Description type="
                                      paragraph" value="
                                      Daily" id="de0006" />
26                        </Answer>
27                        <Answer id="a0006">
```

```
28                                  <Description type="
                                        paragraph" value="
                                        Weekly" id="de0007"/>
29                          </Answer>
30                          <Answer id="a0007">
31                                  <Description type="
                                        paragraph" value="
                                        Monthly" id="de0008"/>
32                          </Answer>
33                          <Answer id="a0008">
34                                  <Description type="
                                        paragraph" value="
                                        Rarely" id="de0009"/>
35                          </Answer>
36                  </Question>
37                  <Question isMandatory="yes" type="
                        selection" id="q0003">
38                          <Limitation minOccur="1" maxOccur=
                                "1"/>
39                          <Description type="paragraph"
                                value="Which car do you dirve?
                                " id="q0003"/>
40                          <Answer id="a0009">
41                                  <Description type="
                                        paragraph" value="BMW"
                                         id="de0010"/>
42                          </Answer>
43                          <Answer id="a0010">
44                                  <Description type="
                                        paragraph" value="
                                        Mercedes" id="de0011"/
                                        >
45                          </Answer>
46                          <Answer id="a0011">
47                                  <Description type="
                                        paragraph" value="Audi
                                        " id="de0012"/>
48                          </Answer>
49                          <Answer id="a0012" isOtherField="
                                yes">
50                                  <Description type="
                                        paragraph" value="
                                        Other" id="de0013"/>
51                          </Answer>
52                  </Question>
53          </Section>
54          <Logic>
55                  <Route type="piping" id="ro0001">
56                          <Condition paperId="p0001"
                                sectionId="s0001" questionId="
                                q0001" isChecked="yes"/>
57                          <Action paperId="p0001" sectionId=
                                "s0001" questionId="q0002" to=
                                "$('pip0001')"/>
58                  </Route>
```

```
59          </Logic>
60 </Paper>
```

4. **Quota and Scoring Logic**

Refer to Fig. Fig. A.130. For instance, questioner sets each answer option has different score.

```
1 <Question isMandatory="yes" type="selection" id="q0001">
2          <Limitation minOccur="1" maxOccur="4" />
3          <Description type="paragraph" value="What types of
               credit cards do you have (Select all that
              apply?)" id="de0001" />
4          <Alignment vertical="stretch" />
5          <Answer id="a0001" score="2.5">
6                                        <Description type="
                                             paragraph" value="Visa
                                             " id="de0002" />
7          </Answer>
8          <Answer id="a0002" score="-2 ">
9                    <Description type="paragraph" value="
                         Mastercard" id="de0003" />
10         </Answer>
11         <Answer id="a0003" score="-1">
12                   <Description type="paragraph" value="
                         American Express" id="de0004" />
13         </Answer>
14         <Answer id="a0004" score="2.5">
15                   <Description type="paragraph" value="
                         Discover" id="de0005" />
16         </Answer>
17                   <Description type="break" value="" id="
                         de0006" />
18 </Question>
```

The total score of this question is 5 points. But each answer option score is different (2.5 point, -2 point, -1 point, 2.5 point). If questioner sets that if this question reach 3 point then jump to Thank you page.

```
1 <Logic>
2          <Route type="skipping" id="ro0001">
3                    <Condition paperId="p0001" sectionId="
                         s0001" questionId="q0001" isChecked="
                         yes" score="3" />
4                    <Action paperId="p0001" descptionId="
                         de0016" />
5          </Route>
6 </Logic>
```

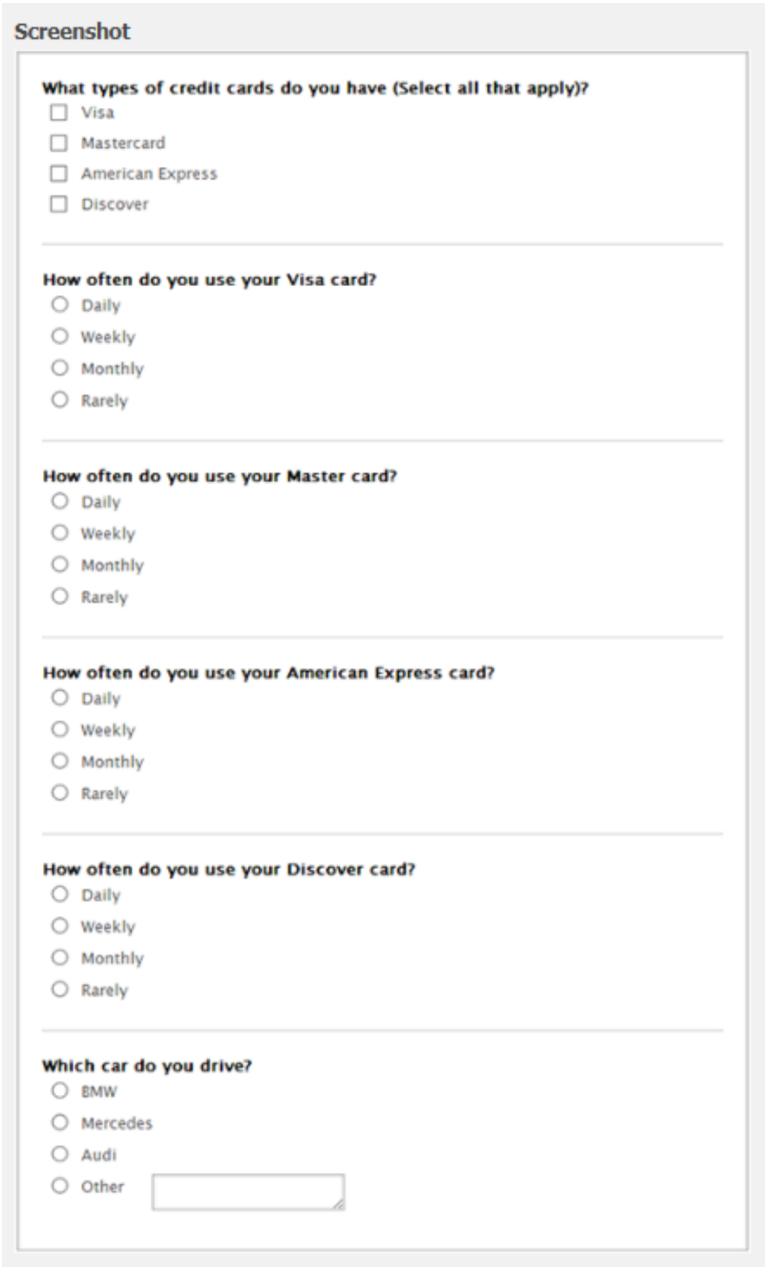

Figure A.130: Screenshot of looping with text piping 2.

Appendix B

Requirements for Systems and Services of E-Questionnaire, E-Testing, and E-Voting

B.1 Requirements for Common Systems

- **Setting up phase**:

  1. **Authority for sponsor and questioner to set an activity**: e-questionnaire, e-testing, and e-voting systems should provide sponsor or questioner the authorities to set an activity.
  2. **Check paper state**: e-questionnaire system should provide questioner to check the paper state such as draft, launched, pause, collected, finished.
  3. **Client-side software connect with database and server**: e-questionnaire, e-testing, and e-voting systems should provide a client-side software connect with database and server to offer service to whom use it (sponsor, executor, respondent).
  4. **Copy paper**: e-questionnaire, e-testing, and e-voting systems should provide authorized questioner to copy paper.
  5. **Create paper/ section/ question/ answer**: e-questionnaire, e-testing, and e-voting systems should provide questioner to create a paper, sections, questions, or answers.
  6. **Delete paper**: e-questionnaire, e-testing, and e-voting systems should provide questioner to delete paper and put it into the trash box (remove permanently or temporarily), sometimes need a password or authority to delete it.
  7. **Delete section/ question/ answer**: e-questionnaire, e-testing, and e-voting systems should provide authorized questioner to delete a sections, questions, or answers.

8. **Device for setting up connecting with software**: e-questionnaire, e-testing, and e-voting systems should provide the device for questioner to set up activity connecting with setting up software.
9. **Edit paper/ section/ question/ answer**: e-questionnaire, e-testing, and e-voting systems should provide authorized questioner to edit a paper, sections, questions, or answers.
10. **Executor task authority**: e-questionnaire, e-testing, and e-voting systems should provide executor tot authorize their tasks.
11. **Generate token with formats**: e-questionnaire, e-testing, and e-voting systems should provide eligible respondents to generate token with formats.
12. **IP address record and limit**: e-questionnaire, e-testing, and e-voting systems should provide questioner to record and limit the respondents' IPs.
13. **Launch paper**: e-questionnaire, e-testing, and e-voting systems should provide questioner to launch paper during a range of time.
14. **Launching device for questioner**: e-questionnaire, e-testing, and e-voting systems should provide record information about launching device for questioner.
15. **Paper and setting database**: e-questionnaire, e-testing, and e-voting systems should provide a database for store paper and setting information for a questioner.
16. **Respondent randomized authenticated token**: e-questionnaire, e-testing, and e-voting systems should provide eligible respondent to randomized authenticated tokens.
17. Search paper e-questionnaire, e-testing, and e-voting systems should provide authorized questioner to delete paper.
18. **Sponsor and questioner authentication**: e-questionnaire, e-testing, and e-voting systems should provide sponsor or questioner to confirm the roles to set an activity.
19. **Stop launching**: e-questionnaire, e-testing, and e-voting systems should provide questioner, executor, or sponsor (authorized person) to stop launch paper when for emergencies.

- **Registering phase**:

1. **Client-side registration software on server**: e-questionnaire, e-testing, and e-voting systems should provide a client-side software for registration for eligible respondents (mainly) and executor, which is connecting with server for registering.
2. **Executor authentication by secrecy**: e-questionnaire, e-testing, and e-voting systems should provide executor to authenticate their role by secrecy (what they know: password).

3. **Executor authentication by token**: e-questionnaire, e-testing, and e-voting systems should provide executor to authenticate their role by token (what they have: ID, randomized tokens).
4. **Registering server**: e-questionnaire, e-testing, and e-voting systems should provide respondent and executor for registering.
5. **Registration database**: e-questionnaire, e-testing, and e-voting systems should provide a registration database to store registration information of respondent and executor.
6. **Respondent authentication by secrecy**: e-questionnaire, e-testing, and e-voting systems should provide respondent to authenticate whether he is eligible or not by secrecy (what they know: password).
7. **Respondent login**: e-questionnaire, e-testing, and e-voting systems should provide authenticated respondents to login and jump to enter answering page.

- **Distributing phase**:

1. **Distribution by channel**: e-questionnaire, e-testing, and e-voting systems should provide a distribution channel for questioner who launch the paper (SMS, WAP, digital TV, Internet, Intranet, Kiosk, postal, telephone, digital storage device, paper, fax, email, website, abroad postal, abroad electronic, abroad other, N/A, other).
2. **Distribution server**: e-questionnaire, e-testing, and e-voting systems shall provide a distribution server to distribute the paper by different channels.
3. **Distribution software**: e-questionnaire, e-testing, and e-voting systems should provide a distribution software for questioner, sponsor, or other kinds of authorized executor.

- **Submitting phase**:

1. **Access channel gateway**: e-questionnaire, e-testing, and e-voting systems should provide an access channel gateway to deal with different channel for submitting (technology).
2. **Answer questions**: e-questionnaire, e-testing, and e-voting systems should provide respondent to answer questions and select options.
3. **Back previous and jump next**: e-questionnaire, e-testing, and e-voting systems should provide respondent to back check previous questions, and jump to next, if the questioner allows (setting this function in setting up phase).
4. **Blind e-papers for anonymity**: e-questionnaire, e-testing, and e-voting systems should provide the respondent to send a blinded e-paper to the submit server (usually called validator) together with some identification and authentication information. This validator inspects the

respondent 's right to submit. If the respondent has the right to submit, the submitting server digitally signs the blinded e-paper and sends it back to the respondent. The respondent now un-blinds the received data and gets a signed e-paper. The he sends this signed e-paper to a second submitting server (often called tallier). Then the tallier knows that the e-paper was sent by an eligible respondent, but he cannot decide which respondent sent the paper.

5. **Client-side software for submitting**: e-questionnaire, e-testing, and e-voting systems should provide a client-side software for respondent to connect with the submitting server for answer questions, submit answers to server, spoil his response, ensure equality and accuracy of presentation of options on any submit device.
6. **Paper database**: e-questionnaire, e-testing, and e-voting systems should provide a paper database to store the answers of the paper related with each respondent.
7. **Error recovery**: e-questionnaire, e-testing, and e-voting systems should provide error recovery on a submitting server to run a self-check before a resuming is possible. In case of irreversible problems the voting server shall prevent a resuming the submitting phase.
8. **Monitor submitting**: e-questionnaire, e-testing, and e-voting systems should provide monitor to monitor the respondents during submitting phase.
9. **Monitor-oriented software**: e-questionnaire, e-testing, and e-voting systems should provide monitor software to monitor the respondents during submitting phase (prevent cheating).
10. **Ping IP address**: e-questionnaire, e-testing, and e-voting systems should provide monitor to monitor the respondents to ping their IP addresses.
11. **Retrieve question**: e-questionnaire, e-testing, and e-voting systems should provide respondent to retrieve questions during this phase.
12. **Submit response**: e-questionnaire, e-testing, and e-voting systems should provide respondents to submit their responses.
13. **Submitting server**: e-questionnaire, e-testing, and e-voting systems should provide submitting sever to
    - be capable of producing comprehensive audit data, to indicate to the executor the number of paper submit so far and its current state;
    - store in the e-ballot box all e-papers submit by eligible respondents during phase;
    - warn the executor if they try to close the activity before the final data;
    - not provide any information about the submitting phase except the current state and the number of papers submit so far;

  - run a self-check before a resuming is possible. In case of irreversible problems the voting server shall prevent a resuming the submitting phase;
  - regularly perform automatic self-checks and report the results to the executors, be capable of performing self-checking;
  - provide executor interface to identify and authenticate, start the phase once, resume the phase after exception, malfunctions, and break-downs, start the counting phase only after having closed the submitting phase, perform self-checks, check that the server has been set up correctly, check the current state, read the audit trails;
  - not reveal the link from the last respondent to his e-paper;
  - remain open a sufficient phase of time to allow for any delay of data transport;
  - support an adequate number of paper;

- **Collecting phase**:

  1. **Collection by channel**: e-questionnaire, e-testing, and e-voting systems should provide a collecting channel for questioner who launch the paper (SMS, WAP, digital TV, Internet, Intranet, Kiosk, postal, telephone, digital storage device, paper, fax, email, website, abroad postal, abroad electronic, abroad other, N/A, other).
  2. **Collection server**: e-questionnaire, e-testing, and e-voting systems should provide collection server to collect submitted data.
  3. **Collection software**: e-questionnaire, e-testing, and e-voting systems should provide a collection software to collect submitted data to analyst or analyzing phase.

- **Analyzing phase**:

  1. **Analysis server**: e-questionnaire, e-testing, and e-voting systems should provide analysis server to analyze the submitted responses, and get the results.
  2. **Analysis software on the server**: e-questionnaire, e-testing, and e-voting systems should provide analysis software connecting with the analysis server.
  3. **Basic analysis**: e-questionnaire, e-testing, and e-voting systems should provide to analyze the responses of the different question types.
  4. **Response database encryption**: e-questionnaire, e-testing, and e-voting systems should provide encrypt the response database away from intruders.

- **Counting phase**:

  1. **Counting software**: e-questionnaire, e-testing, and e-voting systems should provide counting software for eligible counters.

2. **Counts scrutiny to get result and statistics**: e-questionnaire, e-testing, and e-voting systems should provide carefully count to get the results and statistics.
3. **Integrate response counting**: e-questionnaire, e-testing, and e-voting systems should provide to integrate response counting and provide export as some formats.

B.2 Requirements for E-Questionnaire Systems

- **Setting up phase**:
  1. **Export paper by formats**: e-questionnaire system should provide questioner to export paper by some formats (usually .xml, .txt, sometimes .csv, .docx, .xsl).
  2. **Import paper by formats**: e-questionnaire system should provide questioner to import paper by some formats (usually .xml).
  3. **Reminder paper launched**: e-questionnaire system should provide to notice the target respondents that the paper has been launched that you can response it, sometimes also notice them the deadline if the questioner sets it.
  4. **Search paper**: e-questionnaire system should provide questioner to search the papers what they have authority to scan or operate. It also provides the respondents to search it on his homepage and the searched papers should be connect to his account (he must have the authority).
- **Submitting phase**:
  1. **Continue answering**: e-questionnaire system should provide respondents to continue answer questions when they are stopping internally. It exists with the internal stop function.
  2. **Export response**: e-questionnaire system should provide respondents to export his response if the questioner has already set to allow them to do that. And the export response should be downloaded into the local directory by some formats.
  3. **Import response**: e-questionnaire system should provide respondents to import his response if the questioner has already set to allow them to do that. And the import responses should be verified the formats and uploaded form the local directory by the formats.
  4. **Integrate response**: e-questionnaire system should provide respondent to intergrade response automatically when he wants to export the response.
  5. **Internal stop**: e-questionnaire system should provide respondents to internally stop to answer if the questioner has already set to allow them

to do that and the inside the frequency of the stopping. In this function, it lurks a function named continue answering.

- **Analyzing phase**:

  1. **Analyst-oriented software and device**: e-questionnaire system should provide analyst oriented software and device to analyze the collected responses.

B.3 Requirements for E-Testing Systems

- **Setting up phase**:

  1. **Export paper by formats**: e-testing system should provide questioner to export paper by some formats (usually .xml, .txt, sometimes .csv, .docx, .xsl).
  2. **Import paper by formats**: e- testing system should provide questioner to import paper by some formats (usually .xml).
  3. **Reminder paper launched**: e- testing system should provide to notice the target respondents that the paper has been launched that you can response it, sometimes also notice them the deadline if the questioner sets it.
  4. **Search paper**: e- testing system should provide questioner to search the papers what they have authority to scan or operate. It also provides the respondents to search it on his homepage and the searched papers should be connect to his account (he must have the authority).

- **Registering phase**:

  1. **Respondent authentication by token**: e-testing system should provide respondents to register on the system by authenticating his ID information (such as ID card, job card, library card).
  2. **Executor authentication by token**: e-testing system should provide executors to register on the system by authenticating his ID information (such as ID card, job card, library card).

- **Submitting phase**:

  1. **Blinded authenticated token for anonymity**: e-testing system should provide to digitally authenticate a message without knowing the content of the message. The respondent sends a blinded anonymous authentication token (instead of the blinded e-paper) to the validator together with some identification and authentication data. The respondent receives a digital signature from the validator on this blinded token. In the next step, the respondent computes the value for the signed authentication token and sends his data together with his e-paper to

the tallier, which accepts the respondent because of the digitally signed authentication token.

2. **Confirm response**: e-testing system should provide a confirmation to the respondent regarding the status of his paper, at least the information that his e-paper has been successfully stored.
3. **Export response**: e-testing system should provide the respondent to export his response for per question if it is allowed.
4. **Import response**: e-testing system should provide the respondent to import his response by some formats (usually .xml, .txt, sometimes .csv, .docx, .xsl) if it is allowed (for uploaded file question type).
5. **Integrate response**: e-testing system should provide respondent to integrate his exported responses by some formats (usually .xml, .txt, sometimes .csv, .docx, .xsl) if it is allowed.
6. **Internal stop**: e-testing system should provide respondent to internal stop and restart to answer (store previous data) if it is set in setting up phase.
7. **Security software**: e-testing system should provide security software to ensure that the right to cast a paper is dictated by the presence of a token.

- **Analyzing phase**:

1. **Analyst-oriented software and device**: e-testing system should provide an analyst-orientated software and device to analyze collected data (responses).
2. **Trend analysis**: e-testing system should provide a look at data over time for a long-running paper.

- **Marking phase**:

1. **Blink mark response**: e-testing system should provide marker to blink mark responses. It similar with blink analysis. It always mark text question type.
2. **Mark response**: e-testing system should provide marker to mark responses and give the scores.
3. **Marker-oriented software and device**: e-testing system should provide marker-oriented software and device for marker.
4. **Marking server**: e-testing system should provide a marking server to get the response data, identify and authenticate markers, mark the responses and give the score, store the marked data, and send it to counting before deadline.
5. **Marking software**: e-testing system should provide marking software to automatically mark the responses.

- **Counting phase**:

    1. **Mix net for anonymity**: e-testing system should provide a mix net for anonymity as a cryptographic alternative to an anonymous channel. It secures who is communicating with whom and it secures the content of the transferred messages.

B.4 Requirements for E-Voting Systems

- **Setting up phase**:

    1. **Launch option nomination**: e-voting system should provide to launch option nomination, which is a process of approving the options to be presented to voters in a referendum. The options can be a straight choice, e.g. YES or NO, to a single question, or can be more complex involving choices to a number of questions and/or preferences of choice.
    2. **Launch proposal of option nomination**: e-voting system should provide to launch proposal of option nomination, which is a process of approving nominees as eligible candidates for certain positions in an election. A candidate in this context can be a named individual or a party.
    3. **Option database**: e-voting system should provide an option database to store the information for the nominees and the parties.
    4. **Option nomination server**: e-voting system should provide a option nomination server to store the candidate information.
    5. **Option nomination software**: e-voting system should provide option nomination software.

- **Registering phase**:

    1. **Executor authentication by biometric**: e-voting system should provide to identify and authenticate the executors by fingerprints, iris scans, face recognition, voice, manual signature, and DNA.
    2. **Executor authentication by secrecy**: e-voting system should provide to identify and authenticate the executor by password (TAN/ knowledge).
    3. **Executor authentication by token and biometric**: e-voting system should provide to identify and authenticate the executors by fingerprints, iris scans, face recognition, voice, manual signature, and DNA.
    4. **Executor authentication by token**: e-voting system should provide to identify and authenticate the executors by Id card, job card, or library card (ownership).
    5. **Respondent authentication by biometric**: e-voting system should provide to identify and authenticate the respondents by fingerprints, iris scans, face recognition, voice, manual signature, and DNA.

6. **Respondent authentication by token and biometric**: e-voting system should provide to identify and authenticate the respondents by both token (ownership) and biometric (property).
7. **Respondent authentication by token**: e-voting system should provide to identify and authenticate the respondents by Id card, job card, or library card (ownership).

- **Submitting phase**:

1. **Alter message net**: e-voting system should verify the freshness, authenticity, integrity, and format correctness of all messages before processing them.
2. **Auditing interface**: e-voting system should provide an auditing interface to provide the response data for analysis and auditing.
3. **Authenticate the registered respondent**: e-voting system should provide to authenticate and identify the registered respondent whether is the eligible one.
4. **Blinded authenticated token for anonymity**: e-voting system should provide to digitally authenticate a message without knowing the content of the message. The respondent sends a blinded anonymous authentication token (instead of the blinded e-paper) to the validator together with some identification and authentication data. The respondent receives a digital signature from the validator on this blinded token. In the next step, the respondent computes the value for the signed authentication token and sends his data together with his e-paper to the tallier, which accepts the respondent because of the digitally signed authentication token.
5. **Confirm response**: e-voting system should provide a confirmation to the respondent regarding the status of his paper, at least the information that his e-paper has been successfully stored.
6. **Holomorphic secret for anonymity**: e-voting system should provide
7. **Report reason for not voting**: In some jurisdictions, e.g. Australia, where voting is compulsory there is a need to report if an elector has been excused for not voting and what was the reason for the excuse.
8. **Retrieve and amend votes for Pre-ballot box**: e-voting system should provide to retrieve and amend votes before they are counted.
9. **Security software**: e-voting system should provide a security system and sealing mechanism so that trust can be placed in the seal and hence the sealed data. This implies that the seal should be performed as close to the user submission of the vote as technically possible.
10. **Separation of duty for anonymity**: e-voting system should provide a separation of duty approach (also works with at least two submitting servers). One inspecting the right to submit and another one storing

the eligible e-papers. The respondent authenticates himself to the first server. In case that he has the right to submit, he receives a random number generated by this first server. This random number is also send to the second sever but without any information about the respondent ID. Now the respondent uses this random number to authenticate himself as an eligible respondent to the second submitting server. Again this second server can only check whether an eligible respondent sent the e-paper but not who.

11. **Store authenticated paper**: e-testing system should store in the e-ballot box only paper cast from eligible respondents. Any other access to the e-ballot box should be denied.
12. **Store first paper**: e-voting system should store in the e-ballot box at the submitting server only one paper per respondent (the first received paper).
13. **Submitting server**: should communicate only with the authentic and unaltered client-side submitting software.

- **Counting phase**:

    1. **Holomorphic encrypts for anonymity**: e-voting system should provide holomorphic encrypts for anonymity.
    2. **HSM for anonymity**: e-voting system should provide a Hardware Security Module (HSM), which is a tamper-resistant or at least tamper-evident hardware component that can securely generate and store long term secrets for use in cryptography. Generally, it is used to generate a digital key pair without revealing the private key. It can be a function that takes as the input the encrypted e-papers and returns as output the decrypted result, while the decrypted responses are not revealed.
    3. **Mix net for anonymity**: e-voting system should provide a mix net for anonymity as a cryptographic alternative to an anonymous channel. It secures who is communicating with whom and it secures the content of the transferred messages.

- **Auditing phase**: Auditing is the process by which a legal body consisting of election officers and candidates 'representatives can examine the processes used to collect and count the vote, thereby proving the authenticity of the result.

    1. **Audit analysis**: e-voting system should provide audit analysis to audit the total numbers of eligible respondents and issued or qualified ballots provided by the security system with the total number reported. It should also provide trusted data provided by the security process and data provided by the voting process proves that no legitimate votes have been lost by the voting system. It also proves that there is consistency between the number of eligible voters and the spoiled, unspoiled and unused votes as recorded by the e-voting system.

2. **Audit eligible respondents**: e-voting system should provide the total numbers of eligible respondents.
3. **Audit report**: e-voting system should provide the auditing report to prove the authenticity of the result.
4. **Auditing server**: e-voting system should provide an (independent) auditing server for collect and count (specially audit) the result.
5. **Auditing software**: e-voting system should provide auditing software for eligible auditor (mainly) and eligible analyst.
6. **Auditor-oriented software and device**: e-voting system should provide an auditor-oriented software and device independently. It close to the security software and auditing analysis software, as well corresponding device for auditor.
7. **Security Software**: e-voting system should provide a security system and sealing mechanism so that trust can be placed in the seal and hence the sealed data. This implies that the seal should be performed as close to the user submission of the vote as technically possible.
8. **Submitting interface**: e-voting system should provide a submit interface to provide the ballot data for analysis and auditing.

B.5 Requirements for Common Services

- **Paper**:

1. **Answer**: provides a response area with some choices, text area, or others.
2. **Character design**: provides character design and some css style page design for font.
3. **Display ordering**: provides to display answer options in the default order, ascending order, descending order, and random order.
4. **N/A (Not Applicable) option**: provides N/A options that answer will not be factored into the weighted average calculations of question in analyzing. If the respondents using it in a matrix or rating question type, this option will not appear until they select the weights.
5. **Option arrangement**: provides questioner to arrange the option by horizontal/ vertical.
6. **Other field option**: provides selection question type to choose other options, also can fill the value (single text area).
7. **Page design**: provides theme design and some css style page design.
8. **Page**: provides to group the questions to enhance the look of the paper, present with separator sometimes.
9. **Presentation**: provides a message or instruction or heading may also help to visually separate sections.

10. **Question**: provides title and presentation for the question.
11. **Section**: provides to group the questions to enhance the look of the paper.
12. **Sub presentation**: provides a message or instruction or heading may also help to explain the presentation.
13. **Text box arrangement**: provides text box next to question text.
14. **Title**: provide a message for main information.

- **Question types**:

1. **Drag and drop**: provides a certain set of options to be ranked based upon a specific numbering and presented by a dynamic drag and drop.
2. **Drop down list select one answer**: provides a long list of options and ask the respondent select any one from it.
3. **Image chooser**: provides options and images used to get the respondents preference. It provides single selection (radio button), multiple selections (check box), and a rating with a layout of the options. It should provide to upload video. In addition, it should give some tips for the image or video.
4. **Introduction text question with acceptance checkbox**: provides a check box prefixed of a text.
5. **Multiple choices select many answers**: provides preceded checkbox for respondent to choose multiple predefined sets of options.
6. **Multiple choices select one answer**: provides a radio button for respondent to choose one of a predefined set of options.
7. **Open ended text for multiple rows text**: provides a comment box for respondent in a multiple rows text. This kind of question types should provide the number of rows, and prefix and suffix to text box.
8. **Open ended text for single text**: provides a comment box for respondent in a single row text. This kind of question types should provide prefix and suffix to text box.
9. **Open ended text for size-field text**: provides a comment box for respondent in a specific size-field text. This kind of question types should provide the height and the width of the input text size, and prefix and suffix to text box. Sometimes it also provides text box dynamic drag handle at the bottom right corner and presents the pixel of the text box during dragging.
10. **Rank order**: provides respondent to choose and qualify a set of options based on a specific numbering and presented by drop down list after the options with a numeric list.

- **Validation**:

1. **Limit numbers of answer selected**: provides limitation for respondent to limit the number of options. It represent as selection at most/ exactly/least.
2. **Response required**: provides that respondent must respond (default) the question and presents red asterisk before the question.
3. **Text limit for character number**: provides a limit on the number of characters that can be entered in an open ended text question to set up the character limit. This validation must be applied with question required.
4. **Text limit for character style**: provides a validation for text input boxed to validate email address, phone number, and urls.

- **Setting**:

1. **Anonymity**: provides two kinds of ways, firstly, do not record IP addresses of respondents; secondly, do not record the accounts of the respondents by third-party or the systems.
2. **Activation and deactivation**: provides questioner to control the respondents to access to an activity and terminate data collection.
3. **Anti-ballot box stuffing**: provides to prevent respondent to multiple responses.
4. **Automatic numbering**: provides automatic question, section, numbering for matrix style questions.
5. **Back button**: provides to respondent to allow them to back to the previous question.
6. **Copy papers**: provides to copy entire paper including logic.
7. **Countdown timer**: provides questioner to set a time limit on activity to let the respondents have to complete a paper response within a certain period of time.
8. **Display question numbers**: provides to questioner to select whether to display questioners.
9. **Distribution by email**: provides a link via email.
10. **Distribution for Anonymous**: provides to allow anonymous respondents.
11. **Distribution for Track Responses**: provides to allow co-relating or linking individual responses to the respondents.
12. **Done button text**: provides the questioner to define the done button for a specific text.
13. **Email and password authentication**: provides to protect offer limited security as respondents may share the password with others, and let respondents login with email and password.

14. **Exit button**: provides to respondent to allow them to exit to response when they are answering.
15. **Exit url**: provides to respondent to allow them to exit to response and jump to the specific url.
16. **Folders**: provides to be categorized into folders.
17. **IP limitation**: provides the IP limitation for allowing the respondent to multiple responses by record the IP of the respondents.
18. **IP recording**: provides to record the IP addresses of the respondents.
19. **Launch and close time**: provides questioner to set up the launch and close time for activation and deactivation.
20. **Multi-languages**: provides questioner or respondent to design or answer in multi-languages.
21. **Multi-level filtering**: provide to drill down in to data and look at segmented reports.
22. **Next button**: provides to respondent to allow they to see the next question.
23. **Participant ID authentication**: provides to protect offer limited security as respondents may share the password with others, and let respondents login with participant ID.
24. **Participant statistics**: provides to view the overall statistics. It should provide the viewed count, started count, completed count, drop-out count, validation error count, and competition rate.
25. **Password authentication**: provides all the respondents a same password to access the activity. It shall provide the password including in the email invitation.
26. **Password for email detected automatically**: provides to protect offer limited security as respondents may share the password with others, and let respondents to login with different passwords by email invitations automatically.
27. **Real-time report**: provides a quick overview of the real-tile data.
28. **Real-time response collection**: provides a quick overview of the real-time data.
29. **Reminder emails**: provides to sent email batches for whom have used the track url and sent reminders on the setting time.
30. **Report scheduler**: provides a recurring and automated email reports on a periodic basis.
31. **Response Quota control**: provides the respondents to response and in a limited quota.
32. **Save page and continue later**: provides to save the paper midway and continue from where they left at a later time.

33. **Search and replace**: provides a general search and replace for a questionnaire, testing, or voting.
34. **Spelling checking**: provides to check or validate the response is eligible.
35. **SSL secure link**: provides HTTPS paper link.
36. **SSO**: provides single sign-on to allow to authenticate activity account against with third-party system. it should provide SAML, HMAC-SHA1, and DES encrypted.
37. **Time-based data filters**: provides to filter to segment data based on the time ranges.
38. **Username and password authentication**: provides to protect offer limited security as respondents may share the password with others, and let respondents to login with username and password.

B.6 Requirements for E-Questionnaire Services

- **Question types**

1. **Constant Sum**: provides a collection of ratio data to express the relative value or importance of options. It should provide the total points of the total options.
2. **Contact information**: provides consolidated questions asking respondents for their contact information. It shall provide first name, last name, mail address, phone number, and mailing address. It also should provides the validations for the entries, such as mailing address requires a valid zip code, phoned number requires a valid phone number format, and a email address require a valid email address.
3. **Data reference**: provides to collect or validate zip code data against "standardized" database.
4. Date and Time: provides a set of drop down list of date. It should provide setting for select to display date, time, in 12/ 24 hour format, whether default to today. And the ranges of the year.
5. **Download file question**: provides respondent to download the file.
6. **Dynamic multi-tier lookup table**: provides to represent hierarchies of data.
7. **Graphic Rating**: provides the multiple-choice question with different options presented with graphic rate. It provides"thumbs up and down" to select yes and no as the only two options,"star rating" and "face rating" intended to represent sentiments from negative to neutral to positive by 5-point rating scale.
8. **Image hotspot**: provides hotspots selection in an image to let respondent to select.

9. **Likert scale**: provides an opinion list toward any given subject. This kind of question types should provides a scale library (Frequency: Weekly/ Monthly/ Quarterly/ Annually; Yes/No: Yes/ No, Yes /No /Maybe, Yes /No/ NA; Rating Scale: Poor-Excellent, Disagree-Agree, Dissatisfied-Satisfied, 1-10 Rating Scale; Countries: All countries; State/ Provides: countries list).
10. **Matrix multiple choices select many answers**: provides a collection of various individual questions together for respondent to select multiple options for a line marked as checkbox.
11. **Matrix multiple choices select one answer**: provides a collection of various individual questions together for respondent to select one option for a line marked as radio button.
12. **Push to social**: provides respondents to send positive feedback on a survey to social networking sites, and collects comments or reasons for feedback.
13. **Rating Slider**: provides to produce question and answer styles by a slider, sometimes shows the points of the options.
14. **Side-by-side matrix**: provides to collect data on 2 dimensions or more dimensions for the same options.

- **Validation**

1. **Dynamic "parent to child" relationships**: provides to validate the relationship of hierarchies of data, for example, validate a zip code and the corresponding address.
2. **Option quota limit**: provides limitation for quota for each option of the selections.
3. **Response quota limit**: provides limitation for the responses of the paper.
4. **Weighting**: provides a value that gives to each of a number of options to show how important it is compared with the others.

- **Logic**

1. **Automatic redirect**: provides to directly jump to the page of the questioner choice when the respondents finishing answering all the questions. It should provide the finish options, such as urls, link text, and message.
2. **Compound skipping**: provides a combination of multiple options of different questions, if satisfy the options then jump to location. It should provide the criteria name, conditions (question, option, selected), relationship of the conditions (and, or), and setting for logic (if criteria is met jump to, and otherwise jump to).

3. **Extraction question**: provides to display the selected options of a multiple select question as an answer option of the next question.
4. **Looping**: provides to loop option, which allows to dynamically looping through a set of questions base on the response to a multiple-choice question.
5. **Quota control**: provides to control how many respondents you want for each option in a question. In addition, if over limit, the questioner can set a default destination for skipping.
6. **Randomization answer options**: provides respondent to display randomly answer options.
7. **Randomization question**: provides common and block question, and randomly display blocks of questions.
8. **Response quota**: provides a quota for the total number of response of the activity.
9. **Scoring logic**: provides to compute scores or points in real-time. It should to provide the scores for the options, and the branching on score.
10. **Show or hide questions**: provides to show or hide question based on the criteria the questioner defined. It should provide criteria name, conditions (question, option, selected), relationship of the conditions, and setting for logic (if criteria is met, do noting/ show/ hide question).
11. **Skip logic**: provides respondents to answer relevant questions based on their answers. It should provide the default destination to question, terminate paper, chain section, chain paper, thank you page.
12. **Skipping for multiple selections**: provides skipping for multiple selections to corresponding jumping to location, and may also assign piping text for the options.
13. **Text piping**: provides to replace piping text (the options selected by respondent) with the appropriate text to the next question. It should provide the piping text to mark the location of the next question text where to be replaced.

- **Setting**

1. **API**: provides automate customer feedback.
2. **Conjoint analysis**: provides to analyze the results of the conjoint question type.
3. **Consolidate Data**: provides to pair the questions across papers form the drop-downs copy data from a source paper to the current one.
4. **Correlation analysis**: provides a non-parametric measure of correlation, using ranks to calculate the correlation.
5. **Custom variable and mapping**: provides to external variables to connect with email information.

6. **Distribution by live url**: provides a unique link for a paper, and the questioner can post this link to let respondents to response, but the responses collected will be anonymous.
7. **Distribution by SNS integration**: provides to connect with the SNS and share the paper to let the respondents to response.
8. **Dropout analysis**: provides to analyze the drop-out rates for each question.
9. **Email action alerts**: provides to notify immediately when the respondent answers in a particular way to questions.
10. **Gap analysis**: provides to analyze for side-by-side matrix question type.
11. **Heat map analysis**: provides to analyze the results of heat map question type.
12. **Interactive response**: provides to automatic take respondent input without the respondents having to click on continue or submit button on each page.
13. **Multiple criteria segmentation**: provides a mechanism to allow for data segments across multiple questions.
14. **Report sharing**: provides a web link with optional password protection to share the results in public.
15. **SNS authentication**: provides the respondents to connect with SNS count and record the information about UID, full name, locale, gender, link as the data.
16. **Text analysis**: provides to set up text categories with some keywords.
17. **Thank you emails**: provides to sent automatic thank you emails to respondents who complete answering.
18. **Trend analysis**: provides a look at data over time for a long-running paper.
19. **Weighting and balancing**: provides to adjust the data to account for sample bias.

B.7 Requirements for E-Testing Services

- **Question types**:
  - **Connect points**: provides a question type for connecting points from different groups.
  - **Download file question**: provides respondent to download the file.
  - **Formula**: provides a series of letters to support respondent to answer some specific question type for programing, chemical, and mathematics.

- **Image hotspot**: provides hotspots selection in an image to let respondent to select.
- **Matrix multiple choices select many answers**: provides a collection of various individual questions together for respondent to select multiple options for a line marked as checkbox.
- **Matrix multiple choices select one answer**: provides a collection of various individual questions together for respondent to select one option for a line marked as radio button.
- **Upload file question**: provides respondents to upload files with their response. It has the limitation for file types and extension.

- **Validation**:

  1. **Scoring**: provides a score standard for each options or questions.

- **Logic**:

  1. **Automatic redirect**: provides to directly jump to the page of the questioner choice when the respondents finishing answering all the questions. It should provide the finish options, such as urls, link text, and message.
  2. **Randomization answer options**: provides respondent to display randomly answer options.
  3. **Randomization question**: provides common and block question, and randomly display blocks of questions.
  4. **Scoring logic**: provides to compute scores or points in real-time. It should to provide the scores for the options, and the branching on score.
  5. **Show or hide questions**: provides to show or hide question based on the criteria the questioner defined. It should provides criteria name, conditions (question, option, selected), relationship of the conditions, and setting for logic (if criteria is met, do noting/ show/ hide question).
  6. **Skip logic**: provides respondents to answer relevant questions based on their answers. It should provide the default destination to question, terminate paper, chain section, chain paper, thank you page.
  7. **Skipping for multiple selections**: provides skipping for multiple selections to corresponding jumping to location, and may also assign piping text for the options.

- **Setting**:

  1. **Conjoint analysis**: provides to analyze the results of the conjoint question type.
  2. **Consolidate Data**: provides to pair the questions across papers form the drop-downs copy data from a source paper to the current one.

3. **Correlation analysis**: provides a non-parametric measure of correlation, using ranks to calculate the correlation.
4. **Custom variable and mapping**: provides to external variables to connect with email information.
5. **Distribution by live url**: provides a unique link for a paper, and the questioner can post this link to let respondents to response, but the responses collected will be anonymous.
6. **Distribution by SNS integration**: provides to connect with the SNS and share the paper to let the respondents to response.
7. **Dropout analysis**: provides to analyze the drop-out rates for each question.
8. **Heat map analysis**: provides to analyze the results of heat map question type.
9. **Interactive response**: provides to automatic take respondent input without the respondents having to click on continue or submit button on each page.
10. **Multiple criteria segmentation**: provides a mechanism to allow for data segments across multiple questions.
11. **Report sharing**: provides a web link with optional password protection to share the results in public.
12. **SNS authentication**: provides the respondents to connect with SNS count and record the information about UID, full name, locale, gender, link as the data.
13. **Text analysis**: provides to set up text categories with some keywords.
14. **Thank you emails**: provides to sent automatic thank you emails to respondents who complete answering.

B.8 Requirements for E-Voting Services

- **Question types**:

1. **Contact information**: provides consolidated questions asking respondents for their contact information. It shall provide first name, last name, mail address, phone number, and mailing address. It also should provides the validations for the entries, such as mailing address requires a valid zip code, phoned number requires a valid phone number format, and a email address require a valid email address.
2. **Data reference**: provides to collect or validate zip code data against "standardized" database.
3. **Dynamic multi-tier** lookup table: provides to represent hierarchies of data.

- **Setting**:
    1. **Candidate nomination**: provides respondent to nominate or propose candidates.
    2. **Election counting**: provides to count the ballots.
    3. **Participant authority**: provides to authorize each role of participant (especially executor).
    4. **Respondent authentication**: provides to authenticate the respondents by different kinds of methods.
    5. **Respondent registration**: provides to register the respondent information.
    6. **Vote auditing**: provides to audit the ballots and respondents.
    7. **Vote confirmation**: provides to let respondents to confirm their ballots after submitting the ballot (log out cannot).
    8. **Voting anonymity**: provides respondent to answer anonymously. It differs from anonymity represented by not recording IP addresses and account information. It should be aligned with respondent registration.