A Framework for Automatically Generating Quiz-Type Serious Games Based on Linked Data

Wei Shi, Kosuke Kaneko, Chenguang Ma, and Yoshihiro Okada

***Abstract*—Quiz-type serious games are widely used not only for evaluating quiz users' learning effects but also for supporting quiz users' learning activities. However, we suppose that quiz games created by the traditional method have some demerits. First, the storage of the similar quiz questions is extravagant. Second, the choices for each quiz question almost have no or only few changes. Furthermore, current quizzes do not effectively analyze users' activities. For solving the above problems, we propose a new framework which supports to easily create the customized quiz games. The quiz resources are stored as the Linked Data. The linkages among data makes the automatic generation of the choices of each quiz question become possible. Such kind of quiz generation method can realize the wrong choices of the same question are different for each time. Our framework includes two tools. One is to extract and visually represent the schema of the Linked Data. The other is an authoring tool for supporting quiz makers to define a template of quiz pages. The quizzes generated by our framework can collect quiz users' feedbacks and record the users' activities and scores. These collected data will be used for the further analyzation.**

***Index Terms*—Linked data, serious game, quiz, authoring tool, e-learning.**

I. Introduction

In current years, because computers and mobile devices become popular, e-learning plays more and more important role in education. For obtaining better educational efforts, advanced information and communication technology (ICT) such as 3D Animation and VR/AR technologies are wildly used in e-learning. One merit of e-learning is that the e-learning system usually can collect users' log data for the analyzation of users' activities. The analyzed result can be used to improve the e-learning materials.

Serious Games is a kind of e-learning contents. It can both train and test the quiz users. One kind of basic but widely used serious games is the quiz game. However, traditional quiz games are always considered that they have three demerits. First, the questions and corresponding choices of quiz games are usually stored in a database. To build such a database, quiz makers must spend a lot of time. However, these resources are difficult to share and reuse. There are many similar quiz games are created. It means that to create these similar quizzes, a lot of storages and time are wasted. Second, the question's choices are lack of changes. This reduces the training and testing effect of the quiz games. Because many users may answer questions by remembering the positions of right choices or several words in

the right choices to obtain the good scores. Although many quiz game makers add some changes to the quiz games, there changes are still not enough. And more time and storages will be used. Finally, the traditional quizzes normally only provide scores to quiz users as the quiz results. We believe such scores are not enough to evaluate users' learning efforts.

In this paper, we introduce a new framework which supports the creation and customization of quiz games based on the Linked Data. The quiz games created by our framework can solve the above problems. The choices of each question have more changes and the resources for creating these quiz games are stored as Linked Data which supports the easy reuse and share. Furthermore, our framework supports the generation of step questions, which means a question is related and extended from another. For example, we first ask the name of a film, and then ask the director of this file. Then, we can ask the names of other films directed by the same director. Such kind of step questions can help quiz users to understand the knowledge deeply. Last, generated quiz games can collect users' log data for further analyzation to find out the weaknesses of users. The analyzed results will be used to update quiz questions for improving the training effects of the quiz games.

In the following sections, we will explain more details about our framework. The remainder of this paper is organized as follow: Section II introduces the related work; Section III explains the detail of our framework; Section IV shows an example quiz created by our framework. Finally, we conclude this paper and discuss the future work. In the remaining sections, we use the "quiz maker" or the "system user" to indicate the people who use our framework to make their quizzes and use the "quiz users" to indicate the people who use the generated quiz games.

II. Related Work

The resources used to create quizzes in our framework is stored as Linked Data. Linked Data refers to "a set of best practices for publishing structured data on the Web." [1] A data item of Linked Data is defined as a Resource Description Framework (RDF) triple. [2] A triple of Linked Data may link to other triples. Then, we can realize the knowledge discovery through such kind of linkages among tuples. Each

Manuscript received June 30, 2018; revised December 3, 2018. This work was supported in part by the U.S. Department of Commerce under Grant BS123456.

Wei Shi and Yoshihiro Okada are with the Innovation Center for Educational Resources, Kyushu University Library, Kyushu University, Fukuoka, Japan (e-mail: shi.wei.243@m.kyushu-u.ac.jp, okada@inf.kyushu-u.ac.jp).

Kosuke Kaneko is with the Cybersecurity Center, Kyushu University, Japan (e-mail: kaneko.kosuke.437@m.kyushu-u.ac.jp).

Chenguang Ma is with the Graduate School of ISEE, Kyushu University, Fukuoka, Japan (e-mail: machenguanglala@yahoo.co.jp).

RDF triple is composed by a subject, a predicate and an object. Users can use SPARQL to retrieve necessary data. The features of Linked Data allow users to easily share, extend and reuse the data. Currently, many organizations publish their free- accessible Linked Data sets. Different data sets can also connect with each. One of the famous Linked Open Data (LOD) set is the DBpedia (https://wiki.dbpedia.org/). The data in DBpedia is extracted from the Wikipedia (www.wikipedia.org).

Our framework supports to use the linkages of different RDF data to realize the automatic generation of quiz games. As well, some other researchers have already done some similar works. J. Mynarz and V. Zeman [3] introduced the "DB-quiz", that is a quiz generator using the Linked Data retrieved from the Czech and English DBpedia. The quizzes generated by this system are used in a TV show for testing the knowledge coverages of the quiz users. The question format is fixed, and the answer is limited to the labels of RDF data items.

M. Foulonneau [4] discussed a streamline which can generate assessment items (which has the same means as the "questions" in this paper) using the Linked Open Data from DBpedia. However, in this paper, the RDF data used to generate one question seems limited to the triples with the same subject. The linkages among the things are not used effectively.

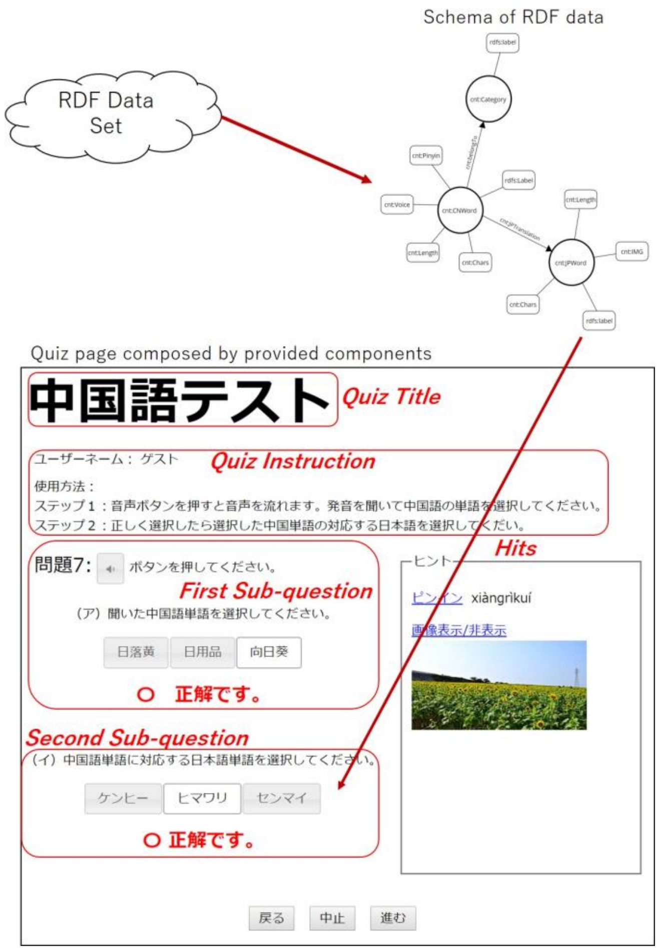
Similarly, D. Liu and C. Lin [5] proposed the "Sherlock", which can semi-automatically generate quizzes using the Linked Data from DBpedia and BBC. In this paper, we think the authors mainly focus on how to "control the difficulty level of the generated quizzes". The templates provided for creating the quizzes are still too basic and may be difficult to improve. The linkages among triples of RDF data are not effectively used as well.

Another problem in these researches is that most quiz makers are considered that they have the knowledge of the RDF data schema. This is also the reason why the above researchers are limited to the special datasets such as the DBpedia. However, there are many different RDF datasets with different schemata. Even some RDF datasets do not have fixed schemata. These papers did not discuss how to use these datasets and how to solve this problem.

III. A Framework for Automatically Generating Quiz-Type Serious Games Based on Linked Data

A. Framework Outline

Our quiz generation framework is composed of two tools. One is an RDF data schema extraction and representation tool. This tool can extract the schema of the RDF data and represent as a graph. By using this tool, users do not need to know the schema of the RDF data which is used to create the quiz. Users can visually manipulate the RDF data schema to define quizzes. The other tool is an authoring tool. This authoring tool provides a set of HTML components for composing a quiz page template. Users need to manually assign the connections between HTML components and nodes in the RDF schema. According to these connections, the system will automatically generate a set of SPARQL queries. Using these queries, our system can retrieve the necessary data from our RDF dataset and provide the values to the HTML components. Then, the dynamic generation of quiz pages will be realized. Fig. 1 shows an overview of using our framework to create a quiz.



中国語テスト

ユーザーネーム：ゲスト

使用方法：
ステップ1：音声ボタンを押すと音声を流れます。発音を聞いて中国語の単語を選択してください。
ステップ2：正しく選択したら選択した中国単語の対応する日本語を選択してくだい。

問題7: ボタンを押してください。

(ア) 聞いた中国語単語を選択してください。

○ 正解です。

(イ) 中国語単語に対応する日本語単語を選択してください。

○ 正解です。

ヒント
ピンイン xiàngrìkuí

Fig. 1. Outline of our framework.

The quizzes generated by our framework have three functions. First, the system will dynamically generate the questions and their choices. Even for the same question, its wrong choices may be different for each time when the question is generated. Second, the generated questions can be persisted into a database for further use. This method can accelerate the speed of the quiz generation and prevent the overload of the server. Third, these quizzes can collect users' log data and feedbacks. By analyzing the log data and the feedbacks, quiz makers can update their quiz games.

The quiz games generated by our framework are Web applications. Each quiz has a server-side application and a client-side application. The server-side application is used to manage the Linked Data, and the client-side application is the quiz pages. The construction of a quiz is shown in Fig. 2. Linked Data used as the quiz resources are stored in the Linked Data server, and the quiz pages are stored in the Application servers. The Linked Data server and the Application server may be the same. The Linked Data also can be obtained from an online server, such as DBpedia.

The application server has two functions. First, it will retrieve data from Linked Data server. Then it uses the retrieved data and the predefined HTML page template to dynamically generate quiz pages. Then the quiz pages will be provided to the users as a Web Application. Through a web browser with an internet connection, quiz users can access this quiz application. Quiz users' log data and feedbacks will be sent back to the application server for further analyzation. Because each quiz game is a web application, it is easy to realize the over-platforms and over-devices. The quiz pages can automatically resize according to the device size which are used to access the quiz application.

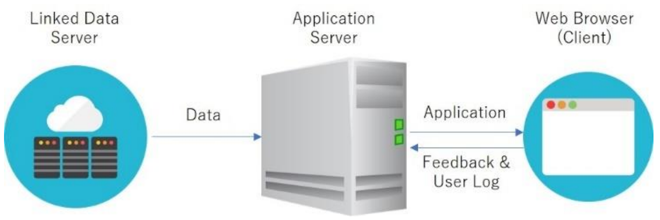


Fig. 2. Construction of the quiz application.

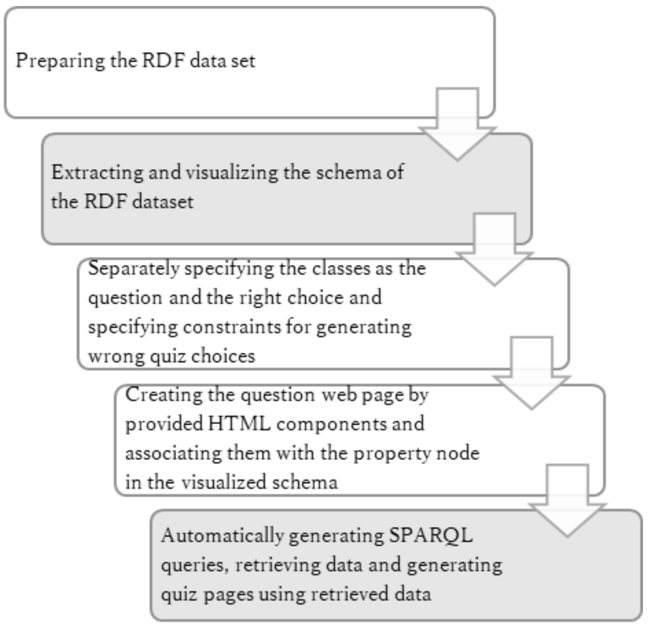

Preparing the RDF data set
Extracting and visualizing the schema of the RDF dataset
Separately specifying the classes as the question and the right choice and specifying constraints for generating wrong quiz choices
Creating the question web page by provided HTML components and associating them with the property node in the visualized schema
Automatically generating SPARQL queries, retrieving data and generating quiz pages using retrieved data

Fig. 3. Workflow of using our framework to create a quiz.

Fig. 3 shows the workflow of using our framework to create a quiz. In this figure, the steps with gray backgrounds will be automatically performed by the system. Normally, users' data may be stored in a relational database. Then, in the step of preparing the RDF dataset, quiz makers may need to use a software to convert the data in a relational database or in a spreadsheet file to the RDF data. Currently, our framework does not include a converting tool, but we plan to propose a tool in future. In most situations, the authoring tool provided by our framework can support users to finish the fourth step without programming. But if a user wants to create his customized quiz, or realize some special functions, he also can write the HTML+JavaScript code to realize them.

In the following sections, we will explain more detail of our framework. In order to explain it clearly, we use a Chinese text quiz game for Japanese students as an example. This quiz has already been published to the students of Kyushu University from June 2018.

In this quiz, we defined a two-step question. Each question page includes two sub-questions. They are used to train the listing and translating abilities of the quiz users. Quiz users first need to listen to a pronunciation of a Chinese word. Then, users need to choose the corresponding Chinese word. If the answer is correct, the second sub-question will be shown. Then users need to choose the Japanese word corresponding to the correct answer of the first sub-question. In the following sections, we basically only use the first sub-question to explain our framework. To simplify the description of the URI in our database, we use the "cnt:" to replace "http://......./CNTestRes/", which is the prefix of the URIs in our database. We also use another two prefixes of the URIs in this paper. We use the "rdfs:" to replace "rdfs: <http:// www.w3.org/2000/01/rdf-schema#>" and use the "rdf:" to replace "http://www.w3.org/1999/02/22-rdf-syntax-ns#". These two prefixes are provided by W3C and widely used in many RDF data sets.

B. *RDF Schema Extraction and Representation*

Comparing to the data in a relational database, RDF data has the more flexible structure. The basic data item of RDF data is a triple composed by a subject, a predict, and an object. Such a structure allows users to update the database easily. But this also makes abstractly representing RDF data schema become difficult. There is no a standard method to represent the schema of RDF data. To solve this problem, we reuse the idea of the ESISW framework proposed by Bin Piao [6] to extract and to visually represent the schema of RDF dataset. To make the schema easy to read, we simplified the graph which used in ESISW framework to represent the schema. Fig. 4 shows the schema of the RDF data for creating our Chinese quiz application.

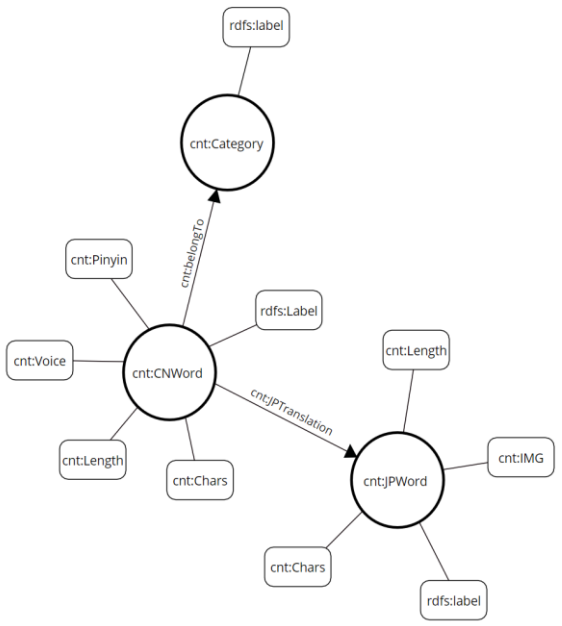


Fig. 4. RDF data schema of the Chinese quiz.

In this framework, the schema is simplified to be composed by two kinds of edges and two kinds of nodes. One kind of nodes is the Class node, another kind is the Property node. The edges in such a graph may be labeled or not. We suppose there is an RDF dataset *R*. As we introduced, the tuple *t* in *R* is identified by a unique URI. We suppose *t* is an instance of class *T*. To generate the schema of *S*, we first extract all the classes in this dataset. A class T_c will be represented as a circular Class node. (e.g. the node "cn:Category" in Fig. 4) Next, we extract all the instances of

T_c and all the tuples whose subjects are these instances. Then, we extract the objects of these tuples and represent them as the nodes. If the value of an object is a fixed value, such as number or text, we add a rectangular Property node to the schema. The label of this node is the predict. (e.g., the node "cnt:pinyin" in Fig. 4) The edge from the Class node T_c to this property node will be unlabeled. If the value of an object is an instance of another Class T_a, an edge between the Class nodes T_c and T_a will be drawn. The label of this edge is the predict of this triple. By repeating these steps, we can obtain the RDF schema.

We suppose the label of a Property node as PN, and it is linked from a Class node with the label CN. Then the Property node is identified as "CN.PN" in our framework. For example, in Fig. 4, the identifier of the node with the label "cnt:Pinyin" is "cnt:CNWord.cnt:Pinyin". We suppose the URI of *t* is U, the object PN of *t* is identified as "U.PN" in our framework.

C. Authoring Tool for Composing Quiz Games

Our framework includes an authoring tool for supporting quiz makers to create quiz games as web applications. This authoring tool is an extension of C. Ma's research [7]. Ma proposes an authoring tool for creating e-textbooks based on Linked Data in this paper. In our framework, we extend the authoring tool to support quiz makers to define the template of the quiz pages.

To simplify users' manipulations, we provide a set of components in our framework. By easily composing these components, quiz makers can define the quiz template. These components are text components, image components, video components, sound components, navigation button components, checkbox components and choice components. Besides these provided components, users can also add their customized HTML elements by directly programming. In our framework, these components are provided as the HTML + JavaScript codes. For each element, users need to determine its position, size and content. The position and size can be absolute values or relative values which can change according to the size of the question page. The content can be an absolute value or a value retrieved from the Linked Data set.

After the quiz maker composes a quiz page template using this method, users need to separately specify the class nodes which are used as the question, right answer and wrong answer. Then users need to associate the components with the Property nodes in the RDF schema through the identifiers of the Property nodes. When a quiz page is generated, the values of corresponding properties will be provided to the HTML components and be displayed on the quiz pages.

D. Automatic SPARQL Generation

After composing a quiz page and associate the components with the nodes of the RDF schema, our system will automatically generate SPARQL query for retrieving necessary data for generating the quiz pages. The amount of the pages of a quiz game can be customized by quiz makers.

First, users need to specify how to generate the quiz questions and their right answers. Supposing we specify a class A as the question and its linked class B as the right answer, the edge between A and B are labeled as L, and the properties "A.P1" and "B.P2" are associated with the HTML components in the quiz page, the system will generate a query as following:

```
SELECT  ?p1 ?p2 ?p2 ?p3
WHERE {
   ?p1 rdf:type "A".
    ?p2 L ?p1.
   ?p1 P1 ?p3.
   ?p3 P2 ?p4.
}
LIMIT 1
OFFSET randomNumber
```

In this query, the variables start with the "?", such as "?p1". It indicates what users want to retrieve. For example, "*?p1 rdf:type "A".*" means to retrieve the subjects of the tuples whose predicts are "*rdf:type*" and objects are "A". The "randomNumber" is a random number generated by the system to ensure a random item is retrieved from the database as the question and the right answer. This query is only used for one quiz page.

If in one quiz page, there are several sub-questions, they can be generated in the same SPARQL query. For example, to generate both the first and second sub-questions of the Chinese quiz, we specify the "CNWord" as the two sub-questions and the right choice of the first sub-question, and specify the "JPWord" as the right choice of the second sub-question. In each quiz page, we use the properties, the images, the pinyin, the pronunciation and the labels of the class "CNWord", and use the labels of the class "JPWord". Then the SPARQL query is as following:

```
SELECT  ?p1 ?p2 ?p3 ?p4 ?p5 ?p6 ?p7
WHERE {
  ?p1 rdf:type "cnt:CNWord" .
  ?p1 cnt:Voice ?p2.
  ?p1 cnt:Pinyin ?p3 .
  ?p1 rdfs:label ?p4 .
 ?p1 cnt:IMG ?p5 .
  ?p1 cnt:JPTranslation ?p6 .
  ?p6 rdfs:label ?p7 .
}
LIMIT 1
OFFSET randomNumber
```

Next step, users need to specify how to generate the wrong choices. As a question's wrong choices, they should have one or more common features with the right choice. Users need to specify one or more constraints to specify such common features. Users can define the constraints through the RDF schema, and then our system can translate these constraints into the SPARQL query. Normally, the wrong choices and the right choices should belong to the same class, but users can specify another class for the wrong choices. To define a new constraint, we suppose that the URI of the right answer is U_r, the wrong choices is U_w, the U_r's property P_{cr} and the U_w's property P_{cw} are used to define the constraint between the wrong answers and the right answer, and the constraint between these two properties is *f*. The constraint is:

$$U_w.P_{cw} = f(Ur.P_{cr}).$$

Currently, our framework supports nine kinds of relationship between the right answer and wrong answers. They are “equal”, “large than”, “less than”, “unequal”, “union”, “and”, “minus”, “contain”, and “not contain”. In future, we will add other necessary computations according to the requests during the developments of quiz games.

As an example, to generate the wrong choices of the first sub-question in our Chinese quiz, we specify the constraints between the right answer and the wrong answers as following:

1) *(Default) The right answer and wrong answers belong to the same class;*

2) *The right answer and the wrong answers have the same length;*

3) *The right answer the and wrong answers should have at least one same character.*

To define these constraints, we need the following nodes in the RDF schema: “cnt:CNWord”, “cnt:Length”, and “cnt:Chars”. We use the label of the wrong answers as the choices. In the constrains definition, we use the “#right” and “#wrong” to indicate the right answer and wrong answer. The class of an instance I is written as I.CLASS. Then the definition is :

```
#wrong.CLASS == #right.CLASS
#wrong.cnt:Length == #right.cnt:Length
#wrong.cnt:Charts ∩ #right.cnt:Charts !== null
```

We suppose the URI of the right answer is *U* and we need *n* wrong answers, then the SPARQL query for generate wrong choices is automatically generated as following:

```
SELECT   ?p1 ?p2 ?p3 ?p4 ?p5
WHERE {
  U rdf:type ?p1.
  U cnt:Length ?p2.
  U cnt:Chars ?p3.
   ?p4 rdf:type ?p1.
   ?p4 cnt:Length ?p2.
  ?p4 cnt:Chars ?p3.
  ?p4 rdfs;label ?p5.
}
LIMIT 1
OFFSET randomNumber
```

This query will be performed *n* times to retrieve necessary wrong answers for the quiz.

For all the constraints, they are divided into the group “necessary” and the group “option”. In the option group, each constraint has a priority. If in the database, there is no enough tuples to generate wrong choices, the constraints in the “option” group with low priority will be ignored. If all the constraints in the “option” group are ignored, and there is still no enough wrong choice, an empty wrong choice will be used in the quiz page. For example, in our Chinese quiz, the second and third constraints are optional, and the priority of the third constraints is lower than the second.

E. Question Persistence

As we explained, for generating one questions with N wrong answers, the system needs to query the database at least (N+1) times. If many quiz pages are requested by different users, the server may overload. To reduce the overload risk of the server, to accelerate the speed of question generation, and to evaluating question’s difficulty level correctly, generated questions will be persisted into a question database. When a new quiz page is generated, the system will select to retrieve an existed quiz page or generate a new quiz page according to quiz makers specification.

F. Properness Guarantee

In our framework, the question and the right choice are generated based on the linkage among the tuples in the RDF dataset. However, the wrong choices are generated according to the constraints defined by users. In some case, the wrong choice may be “right”. For example, in our Chinese quiz, a word whose pronunciation is the same with the right choice maybe treat as a wrong choice. To avoid this problem, quiz makers need to define the constraints more carefully. Besides this method, the system will build a guarantee database to store such kind of errors. When a quiz user finds such kind of mistakes during his answering process, he can report the mistake to the system. The URIs of the question item and the choice item will be stored in the database as a URI pair. When a new question is generated, the system will automatically verify the automatically generated wrong choices using the guarantee database.

G. Quiz Update and Extension

To improve the quality of each quiz created by our framework, our system supports quiz makers to collect two kinds of data from the quiz users. As shown in Fig. 5, we can use two circles to indicate this process.

First, like other applications, the quiz created by our framework can collect users’ subjective feedbacks by performing the questionnaires. The feedbacks mainly used to improve the layout and appearance of each quiz. The feedbacks also contain the bug reports. The updated quizzes will be provided to quiz users again for the further use and evaluation. This process is shown by the left circle in Fig. 5.

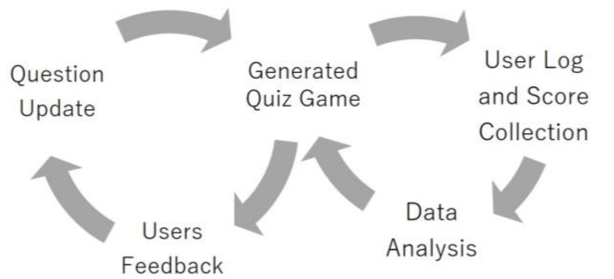


Fig. 5. Two circles to show the processes of updating the quiz.

Next, the quiz also can collect the log data which records both users’ activities and quiz scores. Such kind of data can objectively reflect the users’ knowledge levels and the difficulty levels of questions. As we introduced in Section III.F, all generated questions will be stored in a question database. In this database, each question has two properties, the frequency and the accuracy. The questions will be sorted according to the users’ accuracies. The questions with very high accuracies and high frequency, which means such a question is mastered by most quiz users., will not be provided to the quiz users. The threshold of the accuracy will be manually specified by the quiz maker.

In future, we plan to apply the machine learning technologies in our framework to deeply analyze users' log data. We hope to find out the weakness of each user, and then provide more similar questions to training the quiz users. We also hope to find out the questions which have low accuracies, which means they are difficult for most users to master. Then, in the actual education process, the teachers can spend more time on explaining these questions. These analyzed results will also be used to update the questions of the quiz. This circle is shown as the right circle in Fig. 5.

Our framework also allows quiz makers to extend their quizzes. For example, the second sub-question of each question in our Chinese quiz is extended from the first sub-question. Such kind of extensions can support quiz users' exploration learning. According to the linkage among the data in the RDF dataset, we can provide complex quizzes. In Section IV, we show a pathology quiz. Quiz users need to answer the disease name according to a picture. For extending the quiz, we can request users to answer which viscera or tissue the disease happened. Furthermore, quiz makers may request quiz users to answer what other diseases may happen on the same viscera or tissue. Such kind exploration quiz questions can help users understand a knowledge point deeply and widely. We believe it is very effective in the actual education.

IV. Examples

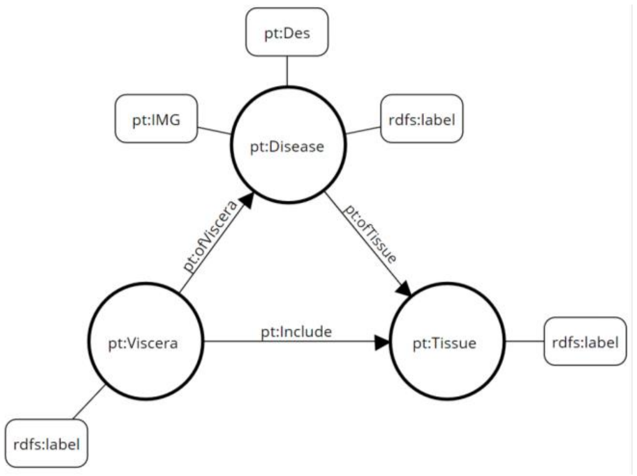
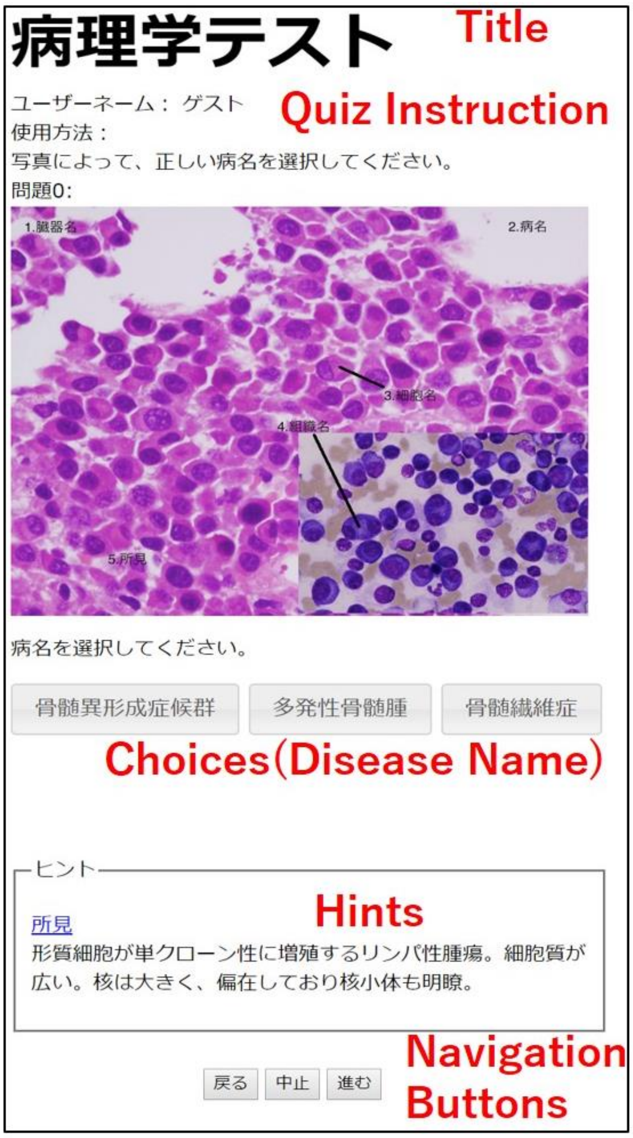


Fig. 6. RDF data schema for creating the pathology quiz.

In this section, we briefly introduce another quiz application, a pathology quiz, created by our framework. We aim to create this quiz for the students of the Medical Faculty, Kyushu University. In this quiz, users need to select the disease name according to the shown picture. We also use the description of this disease as a hit for quiz users. Fig. 6 shows the RDF schema of the data used to create this quiz, and the Fig. 7 is the screenshot of our quiz application.

In this example, we specify the class "ps:Disease" as the question, right answer and the wrong answers. For generating the questions, we use the properties "pt:IMG", "pt:Des", and "rdfs:label" of the class "pt:Disease". Then, the SPARQL query used to generate the question and the right choice is:

```
SELECT   ?p1 ?p2 ?p3 ?p4 ?p5
WHERE {
   ?p1 rdf:type "pt:Disease".
   ?p1 rdfs:label ?p2.
   ?p1 pt:IMG ?p3.
   ?p1 pt:Des ?p4.
}
LIMIT 1
OFFSET randomNumber
```


病理学テスト

ユーザーネーム：ゲスト
使用方法：
写真によって、正しい病名を選択してください。
問題0:

病名を選択してください。

ヒント
所見
形質細胞が単クローン性に増殖するリンパ性腫瘍。細胞質が広い。核は大きく、偏在しており核小体も明瞭。

Fig. 7. Screenshot of the pathology quiz.

Supposing the URI of the disease retrieved by above query is *U*, we define the constraints that the disease of the wrong choices belong to the same class with the disease of the right choice, and these diseases happens in the same viscera. Then, the SPARQL for generating the wrong choices is:

```
SELECT   ?p1 ?p2 ?p3 ?p4 ?p5
WHERE {
   U rdf:type ?p1.
   U pt:Viscera ?p2.
   ?p3 pt:Viscera ?p2.
   ?p3 rdfs:label ?p4.
}
```

LIMIT 1
OFFSET randomNumber

V. Conclusion

In this research, we propose a new framework for automatically generating quiz-type serious games based on the Linked Data. In our framework, it includes an RDF data schema extraction and visualization tool, and an authoring tool. With these two tools, users can create the quizzes which can work as web applications. The questions of each quiz and the choices of each question are automatically generated based on the linkage of the RDF tuples. Quiz makers can easily extend a quiz question, which can support users' exploration learning. The generated quizzes can collect users' feedbacks and log data for further analyzation.

In future, we plan to apply the machine learning technologies to analyze the users' log data. According to the analyzing results, we hope to find out the weakness of each user and to generate the quizzes fitting for different users. We also hope to find out the questions with low accuracies. This result will be provided to teachers for improving the education qualities.

Acknowledgment

This research was partially supported by JSPS KAKENHI Grant Number JP16H02923.

References

[1] W3C. (2016). *LinkedData - W3CWiki*. [Online]. Available: https://www.w3.org/wiki/LinkedData

[2] T. Berners-Lee. (2009). *Linked Data – Design Issue*. [Online]. Available: https://www.w3.org/DesignIssues/LinkedData

[3] J. Mynarz and V. Zeman, "DB-quiz: A DBpedia-backed knowledge game," in *Proc. the 12th International Conference on Semantic Systems (SEMANTiCS 2016)*, 2016, pp. 121-124.

[4] M. Foulonneau, "Generating educational assessment items from linked open data: The case of DBpedia," *The Semantic Web: ESWC 2011 Workshops*, 2011, pp. 16-27.

[5] D. Liu and C. Lin, "Sherlock: A semi-automatic quiz generation system using linked data," *ISWC-PD'14 Proceedings of the 2014 International Conference on Posters & Demonstrations Track*, pp. 9-12, vol. 1272, 2014.

[6] B. Piao and Y. Tanaka, "Interactive framework for exploratory search, integration, and visual analysis of semantic web resources," in *Proc. the 16th International Conference on Information Integration and Web-Based Applications & Services*, 2014, pp. 200-206.

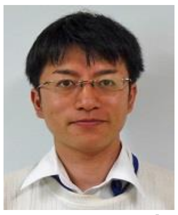
[7] C. Ma, K. Srishti, W. Shi, Y. Okada, and R. Bose "Educational material development framework based on linked data for IOT security," *ICERI2017 Proceedings*, 2017, pp. 8048-8057.

Wei Shi was born at Qiqihar, China. He received a bachelor's degree of applied physics in 2007 from the North China Electric Power University, Baoding, Hebei, China. In 2008, he received a master's degree of computer science from the Hokkaido University, Sapporo, Hokkaido, Japan, and he received a Ph.D. degree of the computer science from the same university in 2015.

He started to work as a postdoctoral researcher at the Research Center for Zoonosis Control, Hokkaido University. Now he is an Assistant Professor of the Innovation Center for Educational Resource, Kyushu University, in Fukuoka, Japan. His research focuses on the information visualization, linked data and e-learning material development.

Ma Chenguang was a master course student of Informatics Department, Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan.

Her research was an integrated framework for educational material development based on linked data supporting learning analytics. It includes development technology such as authoring tools and frameworks based on Linked Data for e-Learning materials of IoT security with learning analytics. She was also interested in serious games of IoT security can be used as educational materials, educational materials based on linked data can be applied to serious game contents as well. Now, she is working as IT specialist in NTT data global solutions regarding SAP HANA, in-memory database.



Kosuke Kaneko is an associate professor of Cybersecurity Center in Kyushu University.

He also works as an adjunct lecturer in Faculty of Art and Design in Kyushu Sango University and Zokei Junior College of Art and Design. He received Ph.D. (Computer Science) from Kyushu University in 2014. He was engaged in research of Educational Technology in Innovation Center for Educational Resource in Kyushu University from 2013 to 2016.

Since 2016, he has worked in current position. His current research interests are cybersecurity and educational technology.

Yoshihiro Okada is a professor of ICER (Innovation Center for Educational Resources), Kyushu University Library, Kyushu University, Japan. He has received his doctorate of engineering from Hokkaido University, Japan in 1993. After that, he worked as a research associate at Department of Electrical Engineering, Faculty of Engineering, Hokkaido University. He was an associate professor of Computer Center, Kyushu University since 1999 and an associate professor of Graduate School of Information Science and Electrical Engineering, Kyushu University since April 2000. He obtained his current position in 2013 and he has been a director of ICER since April, 2015 and a vice-director of cybersecurity center, Kyushu University since 2017. Currently, his research interests include 3D graphics, HCI, VR/AR, network collaboration, educational material development and cybersecurity.