International Journal of Innovative
Computing, Information and Control
Volume 11, Number 2, April 2015

 ISSN 1349-4198
pp. 743–757

PROPOSAL FOR E-LEARNING SYSTEM WITH EXPANDABLE AND DEVELOPABLE FUNCTIONS

Hiroshi Mabuchi and Toshiki Satoh

Faculty of Software and Information Science
Iwate Prefectural University
152-52 Sugo, Takizawa, Iwate 020-0693, Japan
mabu@iwate-pu.ac.jp; g231m018@s.iwate-pu.ac.jp

Received May 2014; revised September 2014

Abstract. *Although e-learning systems are used by a wide number of learners and teachers, it is not necessarily true that all the functions they need are provided. In this study, we will discuss how to develop an e-learning system which will allow learners and teachers to create system functions and then share them by incorporating these functions into the system. E-learning systems are implemented through web applications. We will explain the reason why it is difficult to create these web applications and discuss a specification program which abstractly describes a common gateway interface (CGI) program, HTML and database access – the elements of a web application. This specification program is described by an ET program, one based on the theory of equivalent transformation (ET) models, and expresses the web application's parallelism through the ET program framework. We will also explain how to incorporate a specification program into an e-learning system by transforming the program into an actual web application. Compared to conventional methods in which CGI programs, HTML and database access are directly scripted, the method proposed in this study can clearly describe with a lower amount of code and without requiring background knowledge.*
Keywords: E-learning system, Web application, Equivalent transformation program, Database

1. **Introduction.** Recently, with the growth of computers and the Internet, it is becoming more popular to use information technology for university coursework or for companies to use it for in-house training. A specific example of those is an e-learning system. E-learning forms include computer based training (CBT), web based training (WBT) and distance learning (DL) [1, 2, 3, 7, 8, 9, 11, 12, 13, 14]. In particular is WBT, widely used because multiple learners from a variety of locations can receive their education through the Internet. A characteristic of today's e-learning is that communication is possible between teachers and learners or learners and learners thanks to the advancement and pervasiveness of communication technologies.

E-learning system functions are provided by system developers and, through the use of these functions, learners are able to learn while teachers are able to provide the course materials and grade scores. However, it is not always the case that learners and teachers are provided with the functions they need. Some learners may need to analyze their academic results and display them in a uniquely represented manner in a table. It is not so easy to create a function because it is difficult to immediately come up with the demanded functions due to the high cost (choosing, designing, incorporating and testing each function) of system development [19, 20].

And, e-learning systems are implemented through web applications. A web application is comprised of a common gateway interface (CGI) and database and is realized

through network communications made via browser. Creating a system function requires understanding of CGIs, HTML documents to be sent to the browser and database access; because specialized skills are needed for its development, not everyone is capable of simply creating a function due to the level of difficulty [19, 20].

In this study, therefore, we will examine a method by which system functions can be created easily and propose an e-learning system which will allow learners and teachers to create the unique functions they need. First, based on the characteristics of web applications, we will find what causes difficulty when creating functions and then propose a method that expresses the CGI, HTML and database – the elements of a web application – in a single model.

An equivalent transformation (ET) programming language will be used for the modeling language [4, 5, 15, 16, 17]. This is because the non-deterministic computation concept of the ET computation model [18] is suitable for modeling a web application's parallel world and for modeling the database in a simple and effective manner. Using these models, we will explain a method by which e-learning system functions can be easily created.

Also, we will provide an environment where the functions, which are created by users in the proposed method, are actually used in e-learning and explain a method by which learners and teachers can share the system. Through this, we aim to reduce the cost required to create e-learning system functions, a cost which was previously very high.

2. Study of ET Model and ET Model-Based E-Learning Systems.

In this section, we will provide a general explanation of the ET model [4, 5, 15, 16, 17] and make evident the value of ET model-based e-learning systems.

2.1. Equivalent transformation.

An equivalent transformation (ET) is the operation in which when a problem prb is transformed into a simplified problem prb', the original meaning of prb is preserved within prb'. That is, when the meaning of prb is $M(prb)$ and the meaning of the simplified problem is $M(prb')$, the relationship $M(prb) = M(prb')$ is established. In the ET theory, computation is performed through equivalent transformation, and the given problem is solved by repeating equivalent transformations with the original meaning preserved.

2.2. Computation by equivalent transformation.

Computation by equivalent transformation is performed in the following sequence [15].

1. A user gives the system a single transformation atom and sets the transformation clause as follows.
   (ans <transformation atom>) ⟵ <transformation atom>
2. The system applies the equivalent transformation (ET) rules, previously given by the user, to the body atom in the transformation clause.
3. 2 is repeated until the body atom in the transformation clause becomes empty or there are no applicable rules.
4. The transformation result is output.

2.3. ET rule.

In the ET rules, there is a D rule (D stands for deterministic) and an N rule (N stands for non-deterministic) [15, 21].

In the D rule, rules that are first described are applied before rule(s) that are described later. Also, in terms of the order by which body atoms are applied, atoms that are first described are applied before atom(s) that are described later. The D rule's characteristics are shown in the following.

- There is only one pattern of order by which rules are selected and only one pattern of order of atoms to which the selected rules are applied (Deterministic).

- Since only one pattern of order for rule selection is specified, rule priority does not exist.
- There is one body or less.

Unlike the D rule, the N rule has no defined pattern of order by which rules are selected or of atoms to which the rules are applied. This is the language to describe non-deterministic algorithms. The N rule's characteristics are shown in the following.

- The order by which rules are selected and the order of atoms to which the selected rules are applied is non-deterministic (sequence determined by the system).
- Priority can be assigned to each rule and the order of the rules can be controlled to some extent based on this priority.
- Rules which contain multiple *Head* (multi-head rule) can be described.
- Rules which contain multiple *Body* (multi-body rule, i.e., splitting rule) can be described.

2.4. **Significance of the proposed system compared to conventional e-learning systems.**

2.4.1. *Conventional e-learning systems.* This section describes characteristics and problems about conventional e-learning systems.

[Characteristics]

Currently, in the most commonly used e-learning systems, such as Moodle, system developers provide e-learning system functions and learners and teachers use the functions they are given [19, 20].

Moodle can be utilized as an e-learning system by installing its core code, a set of coded basic functions, to the server. Functions not originally implemented into the core code can be either installed by users through Moodle plugins or added to the system by system developers.

[Problem areas]

It is not always the case that learners and teachers are provided with the functions they need. When they are not provided, system developers, learners and teachers are able to add unique or customized functions. However, e-learning systems such as Moodle are realized through web applications and creating a system function requires understanding of CGIs, HTML documents to be sent to the browser and database access; because these specialized skills are needed for development, it is difficult for learners and teachers to simply create functions. Also, for system developers, there is a high cost in choosing, designing, incorporating and testing each function, so it is difficult to immediately provide requested functions.

Furthermore, even if functions are added by system developers, learners or teachers, it does not necessarily mean that the correctness of the entire e-learning system will be guaranteed (that is, the system properly operates) with these added functions.

2.4.2. *Characteristics of ET-based e-learning system and its significance as compared to conventional e-learning systems.* To overcome problems described in Section 2.4.1, in this study, functions will be created or added only through the use of ET model-based programming language [4, 5, 15, 16, 17, 21].

[Characteristics of ET-based e-learning system]

Because e-learning systems are realized through web applications, the non-deterministic computation theory [10] of the ET model is suitable for modeling a web application's parallel world and for modeling the database in a simple and effective manner. Furthermore, because independent ET rules are used to create functions under the ET model, the correctness of the entire system can be guaranteed with any rules (functions) added to the

system as long as the rules are correct. Therefore, it is not necessary to review the entire system such as being required with Moodle.

[Significance and advantages of the proposed system compared to conventional e-learning systems]

(1) It places less of a burden on system developers. With Moodle, only users with system developer access can install new functions. Its system developers, therefore, must find from among Moodle plugins, or create on their own, the functions that many teachers request. On the other hand, because the proposed system makes it possible for teachers to create new functions which suit their needs, the only requirement of system developers is to incorporate these new functions into the system.

(2) Using a single programming language, anyone can, with relative ease, create and add functions to a system and there will be guaranteed correctness of the entire system following the implementation of the new functions. Although teachers and learners can create/add functions with Moodle as well, doing so requires them to have specialized knowledge and skills related to a variety of programming languages. Also, adding functions to the system will compromise the guaranteed correctness of the entire system and, as a result, the system could fail.

(3) The components by which functions are composed have a low level of granularity. Under Moodle, functions must be installed in large units, which means that if part of an existing function needs to be modified, or a new feature is to be added, the entire function must be modified. On the other hand, the proposed system treats functions as a set of fine, specific operations (ET rules), so if part of a function is to be modified or added, only the corresponding ET rule needs to be modified/added.

3. E-Learning System and ET.

In this section, we will discuss web applications used to realize e-learning systems, ET-based CGI programming method and database access method. Also, we will explain what causes difficulty when creating web application functions.

3.1. Web application and CGI.

Web applications are realized via network between client and server. Their elements include browser, CGI, database and network [4, 5].

The browser displays data sent from the server and is also the interface for the client to send data and requests to the server. CGI is the mechanism by which the server activates programs in response to client requests, accesses the database and dynamically generates and returns documents [6].

CGI programs process data sent from the client, access the database and conduct the communication. Based on the CGI programs' processing, HTML documents are dynamically generated and, by returning these HTML documents to the client via the server, dynamic web pages can be displayed. Although CGI uses any programming language, the e-learning system used in this study uses the ET programming language to describe the CGI program. The following describes CGI's basic points and the programming methods for CGI based on the ET model.

3.2. Basic CGI programming.

This section describes basic CGI programming. The basis of CGI programming is to return HTML documents to the client. The generated HTML documents are sent back to the client through standard output. At the start of output, it is necessary to output a content-type header which indicates the type of output. This is to specify, between the server and browser, the data format; with regular files, the data format is specified through its extension but with web applications, the data format is specified by MIME type. This MIME type information is indicated in the content-type header. MMIME type formats are as follows.

TABLE 1. Most common MIME types

File format	Common extension	MIME type
Text file	.txt	text/plain
HTML document	.htm .html	text/html
XML document	.xml	text/xml
JavaScript	.js	text/javascript
GIF image	.gif	image/gif
JPEG image	.jpg .jpeg	image/jpeg
MPEG file	.mpg .mpeg	video/mpeg

3.3. **Data transfer method.** CGI performs computation based on user input data, accesses the database and dynamically generates the next web pages that are to be displayed. This section describes data transfer between the web browser and server.

3.3.1. *Sending data.* The information that is required to send data includes "what to send", "where to send" and "how to send". There are several methods by which CGI sends client input data to specified programs on the server and the following describes three of those methods: "send by form", "send by anchor" and "send by script".

send by form:

In HTML, a structure called "form" is included to achieve data transmission. HTM L's "form" presents the method (control) to input the data and "what to send", which is a section that consists of a label and explanation for the input data. This section of the form determines "where to send" and "how to send".

Form elements have two major attributes. One is an action attribute that specifies "where to send", that is, specifies the program to receive the data. The action attribute's value specifies the recipient program's uniform resource identifiers (URI) and sends the data. The other is a method attribute that specifies "how to send", that is, specifies the data transmission method. In the method attribute, there are two methods, "post" and "get", and if the attribute is left out, it will be get. The difference between "post" and "get" is explained below.

- Post is used to "post" data such as writing messages or registering new data. Because it is transferred to standard output, it will not be displayed in the URI and does not have a limit on data length.
- Get is basically used to extract something. With get, a query is displayed as part of the URI and is transferred to CGI via the server's environmental variables, so there is a limit on data length.

The data to be transferred becomes "name" and "value" paired format. The input element consists of methods to receive input from users and provides these various input methods by changing the type attribute. The following is a list of input element type attributes.

send by anchor:

This is a method by which the HTML <a> tag's href is described with a destination script that is specified by URI and data. When a user clicks a link, data is sent to that script.

send by script:

This is a method by which data are transmitted by JavaScript. JavaScript is a script language suitable for web browser applications and is able to dynamically change web pages displayed on a browser or checks the values that have been entered into forms.

TABLE 2. Input element attribute

Attribute	Explanation
text	row text input field
password	password input field
checkbox	checkbox
radio	radio button
button	general-purpose button
submit	send button
reset	reset button
file	select a file to send
image	make a button from specified image
hidden	data to be sent without being displayed

3.3.2. *Receiving data.* To receive data, there are two methods: "get" (Figure 2) and "post" (Figure 1). The form in which the data was sent will be recorded in environmental variable REQUEST_METHOD, the receive method changing according to that value. The following describes how to receive data in an ET program. When sent by post, the data length set in environmental variable CONTENT_LENGTH is sent to standard input and data that will be read from the standard input will only be for the length set in CONTENT_LENGTH. Because the data is URL encoded, it has to be decoded by (cgi_decode/2). Once decoded, the transferred data can be generated in list format. The program received by post is as follows.

```
1 (as (cgi:read *data)
2     (getenv "REQUEST_METHOD" "POST")
3     (getenv "CONTENT_LENGTH" *len)
4     (atoi *len *i_len)
5     (stdin *i_len *code)
6     (cgi_decode *code *data) : )
```

FIGURE 1. Receiving data by POST

When data is sent by get, the data is set in environmental variable QUERY_STRING and that data is decoded by (cgi_decode/2). The program received by get is as follows.

```
1 (as (cgi:read *data)
2     (getenv "REQUEST_METHOD" "GET")
3     (getenv "QUERY_STRING" *code)
4     (cgi_decode *code *data) : )
```

FIGURE 2. Receiving data by GET

The received data becomes a "name" and "value" paired list. For example, when data is sent using the anchor,

```
<a href="sample.eti?name1=bar1&name2=bar2">Next page</a>,
```

the received data is generated in the following format.

```
((‘‘name1’’ ‘‘bar1’’) (‘‘name2’’ ‘‘bar2’’))
```

Because various parts of the script's contents refer to the generated data, within the CGI program, which utilizes global data domain, there is an associative array which can be accessed from anywhere. The following shows the program using the associative array.

```
1  (include “map.etc”)
2  (include “cgi-lib.etc”)
3
4  (as (main)
5      : (ReadPhase    IN)
6        (map:find     IN “name1” *data)
7        (print *data))
```

FIGURE 3. Global data domain

Figure 3 is explained here.

- First and second rows: Libraries are read.
- Fifth row: Data that was sent to global data domain IN are read as associative arrays.
- Sixth row: Data from global data domain IN that has the name name1 are substituted with the variable *data.

The rule to access the data set in global data domain is as follows.

- (map:find *sym *name *value)
  Data that has *name, from which the value *name will be acquired as *value from the domain specified by the variable *sym, must exist or the rule will fail.
- (map:find2 *sym *name *value)
  If data that has *name, from which the value *name will be acquired as *value from the domain specified by the variable *sym, does not exist, an empty string will be returned.
- (map:search *sys *name)
  Data containing *name should be checked for in the domain specified by *sys.
- (map:erase *sys *name)
  Data containing *name that exists in the domain specified by *sys should be deleted.

3.3.3. *Database access.* This section explains database access which is absolutely imperative for web applications.

SQL is used for database access. SQL stands for structured query language and is a database language that manipulates and defines data in a relational database management system (RDBMS).

SQL consists of three major types: data definition language (DDL), data manipulation language (DML) and data control language (DCL). DDL contains CREATE statements which define tables and constraint conditions and DROP statements which delete tables. DML contains SELECT statements which extract records, INSERT statements which insert records to tables, DELETE statements which delete records and UPDATE statements which update fields of specific records. DCL contains BEGIN statements which declare the start of transaction processing, COMMIT statements which dictate the completion of transactions and ROLLBACK statements which cancel transactions.

When using software to manipulate the database, SQL statements should be generated by the program and then issued to the RDBMS to execute the operation.

3.3.4. *Causes of difficulty in creating web applications.* Conventionally, creating web application functions was viewed as creating CGI scripts on the server and as a result, the general architecture was not taken into account. The reasons for which it is not possible to easily create web applications are as follows.

- It is hard to grasp the relationships between browsers, CGI and networks
- Complexity of HTML description for browser display
- Complexity of data transfer/acquisition between pages (e.g., form, hidden tag)
- Difficulty in managing user IDs and passwords at login
- Complexity of database access (SQL statements)

That is, web applications contain a mix of different models such as CGI, HTML and database access, and the reality is that creating web application functions is impossible unless all those models are understood. Web applications are difficult to understand because of this reason and it is challenging for learners and teachers to create functions.

4. Specification Program.

This section describes a specification program that models web applications.

4.1. Rule-based descriptions of web application general architecture.

Web applications are tools used to transfer data between browsers and CGI via networks. Browsers display data, input forms and buttons on screen as received from the server while the server accesses the database, processes data received from the browser and generates data to send back to the browser. Networks conduct the transfer of data between them. The essential elements in describing the general architecture are as follows.

- Screen display on browser
- Database access and data processing on server
- Connection via network

N rules of ET rules are used to describe the specifications [15, 21]. The general architecture of web applications can be described using ET rules.

Figure 4 illustrates the specification program which is described in this format. In the proposed specification program, the *Cond* atom line indicates the browser's screen display and the *Exec* atom line indicates database access on the server as well as data processing. Also, the transformation from *Head* atom A to *Body* atom B as well as the transfer of variables shows the network connection and data transfer. This makes it possible to easily capture an overall picture of a web application.

Furthermore, *Cond* atom and *Exec* atom are described through the use of the D rules of ET rules [15, 21]. The *Exec* atom's processing represents the server's processing, so conventional CGI programs are equivalent to the processing of *Exec* as described by the D rules.

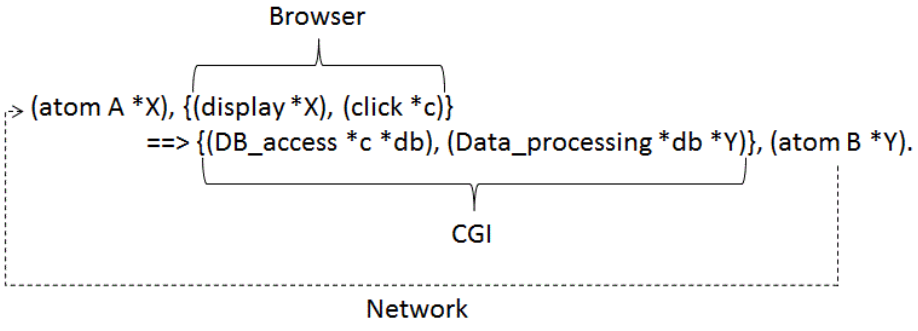

FIGURE 4. Specification program

From this, database access and data processing to send back to the browser are described. Hereafter, detailed elements are presented.

4.2. **Description of elements essential to function creation.** We have described the general architecture of web applications such as browsers, CGI and networks. Furthermore, descriptions of the elements essential to creating functions are needed. The following are needed for web applications.

- HTML tag description
- User information (ID, password) description
- Data transmission (form, hidden) description
- Database access description

That is, to create a web application, it is crucial to understand a variety of models such as CGI programming, HTML and database. And so, these will be presented as a single database model.

4.2.1. *HTML tag description.* HTML is used by browsers to make display possible for clients. HTML is a markup language used for describing web documents. Tags are used for description, and the tag structure is used to describe logical structure and display.

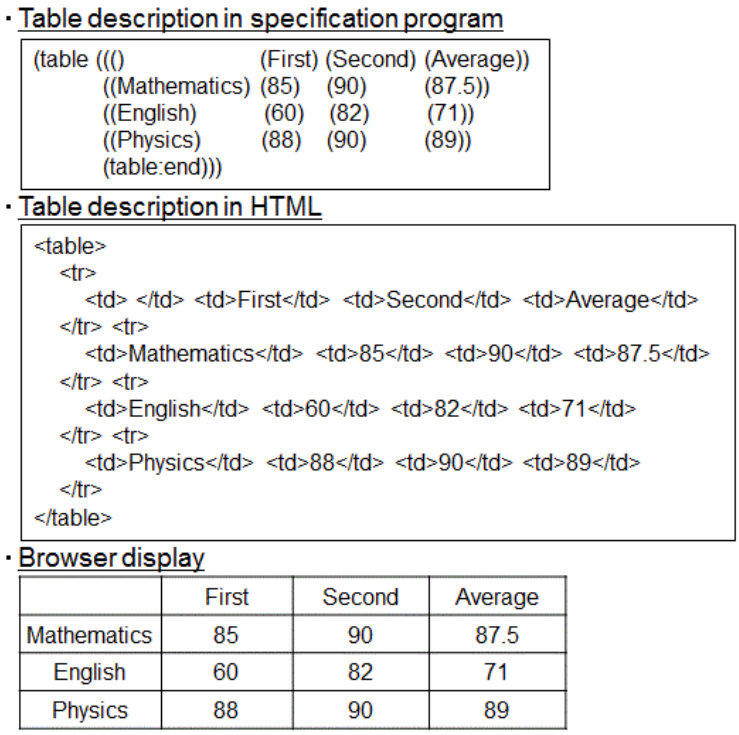

FIGURE 5. Example of table tag description

To understand this HTML and to write texts correctly, it is important to understand the tag structure. And so, the HTML tag structure will be expressed with a list structure handled under ET rules. As an example, a <table> description is described in Figure 5. The specification program's *table* description and the HTML tag's <table> description have the same meaning. Writing the table structure in list structure in the specification program gives an intuitive description. Also, display data and display format are separately described through specifying the display format in the list's first element and the data to be displayed in the second element. Similarly, for the <ul> tag,

(ul ((bulleted 1)
(bulleted 2)))

performs the same as

bulleted 1
bulleted 2,

and as a result, the same content can be described in this manner.

As illustrated here, by expressing the same meaning through list structure, the ET rule model lets HTML be expressed even without knowledge of HTML.

4.3. **User information and data transmission description.** Transmission of user information (user ID, password) and data (form, hidden tag, etc.) are conducted between browser and CGI and realized via network. As previously described in Section 4.1, under the specification program, the transformation from *Head* atom A to *Body* atom B as well as the transfer of variables shows the network connection and data transfer. That is, the described data transmission is realized by the transfer of variables from one atom to another atom. Also, user information constantly travels back and forth between the browser and CGI while e-learning system functions are in use and will not be lost while in transit. During transformation of *Head*_atom to *Body*_atom, user information is also constantly transferred at the same time, so a description of user information is not needed.

(*Head*_atom *variable), { ··· }
$\Longrightarrow$ { ··· }, (*Body*_atom *variable).

Therefore, descriptions of user information and data can be expressed in the specification program's transformation of *Head*_atom to *Body*_atom and the transfer of *variable.

4.3.1. *Database access description.* SQL is commonly used for database access. As shown in Section 3.3.3, for SQL descriptions, it is essential to understand database structures, table names, line names and the relationship between tables, so it may be hard for learners and teachers to make database access descriptions.

Here, an ET program is used for database representation and access. The database concept can also be addressed with the ET program.

Databases can be modeled through definite clauses, and the ET program is one which manages definite clauses. The database's table is equivalent to the ET program's atoms,

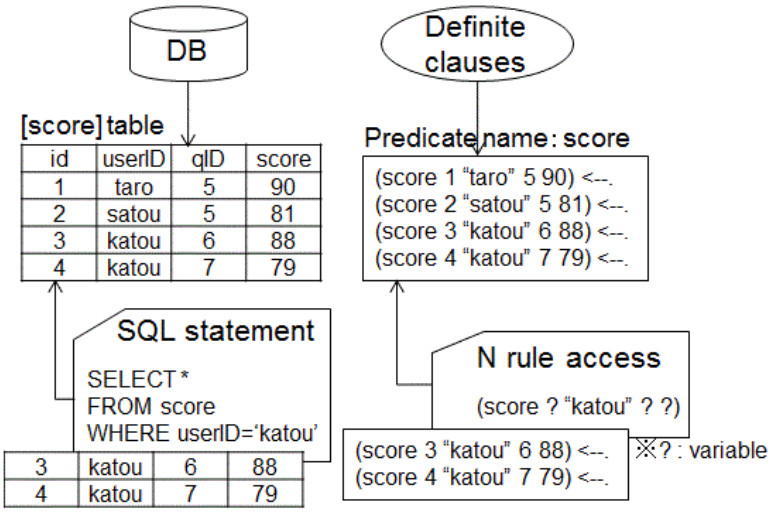

FIGURE 6. Database and definite clauses

and queries to database (data acquisition) are equivalent to "calling N rules" (Figure 6). Therefore, access to the database, which was modeled by definite clauses, is possible by getting and using N rules under the specification program. Although efficiency decreases with ET programs compared to SQL statements, it is easier to describe what is being requested.

As shown above, under the single model of ET rules, the specification program is capable of describing e-learning system functions in a manner that is simpler and clearer. Unlike actually creating functions, it describes the web applications' general architecture and essence without requiring complex scripting. In the next section, we will register a specification program into an e-learning system and propose a method that will actually run the program.

5. **Proposed System.** In this section, we will explain the system that runs the specification program the same as actual CGI, describe the mechanism that releases and shares the specification program and describes the mechanism that incorporates the specification program into an e-learning system.

5.1. **Specification program operation.** To run a specification program, that specification program must be registered first. Registration of the specification program is conducted on the e-learning system. Items which need to be registered are the specification program and function title description. These will be automatically registered to the database when the registry button is pressed. The function that registers the specification program is called registration CGI.

For the specification program to run on the e-learning system, the specification program must be transformed into a format that actually functions. Also, the environment that will enable the transformed program to function must be prepared. This transformation and operation is conducted on the e-learning system, automatically running when the function is utilized by a user. The mechanism that transforms the specification program and the one that actually runs on the e-learning system, as described above, is called interpreter CGI. In the next section, the transformation of the specification program and the operation of the transformation program are explained.

5.1.1. *Transformation of specification program.* To actually run the specification program as a web application, the specification description transforms the program into a simpler expression.

Items that need to be transformed are listed in the following.

- HTML description
- User information and data transfer description
- Database access description

The transformation is something that the system automatically executes, not something that must be executed by the learners or teachers who created the specification program.

With regard to HTML tags, as shown in <table> description example in Figure 5, they were realized through a simpler description method which still contained the same meaning of the HTML description. Therefore, the specification program can be transformed while preserving the meaning of each HTML tag.

As described in Section 4.3, user information (user ID, password) and data transmission (form, hidden) are described by the transformation of $Head$ atom to $Body$ atom and the transfer of variables. Therefore, the program is transformed into a format that adds user information during atom transformation and adds the <form> tag and <hidden> tag during variable transfer.

With regard to database access description, it is described through the ET program as previously mentioned in Section 4.3.1. Therefore, access to the database, which was modeled by definite clauses, is possible with the ET program. Also, automatic generation of SQL statements is possible with the ET program. From this, faster access is possible.

5.2. **Transformation program operation.** After transforming the specification program, it will be of no use if an environment in which it can function is unavailable. Therefore, the operating environment to which the transformed program can be loaded and function within will be provided. This operating environment is like infrastructure and is where the transformed specification program is loaded, in the same manner as data, and executed.

From the specification programs that are registered by registration CGI, a single program that is to be executed is chosen by the user. Then, the selected specification program is transformed, and the transformed program is loaded for computation. The results are then sent back to the user. That is,

$$\text{Transformation} \Longrightarrow \text{Action} \Longrightarrow \text{Result returned}$$

is what interpreter CGI will carry out.

5.3. **Releasing and sharing.** When functions are created with specification programs, there may be cases where the person who created the function wants to share it with classmates or cases where teachers have created a function and want their students to use it. Also, there may be cases where a person wants a new function but it has already been created by someone else. Furthermore, there may be cases where a person wants to take a function that someone else created and customize it for their own specific use.

Therefore, a mechanism that allows for the release and sharing of programs was created. At the discretion of the person who created it, created functions can be released to the entire class. The release of functions is done at the specification program level, and the copying and modifying of another person's program is also possible. The reason why sharing is done at the specification program level, not with a program whose specifications have been transformed, is because specification descriptions make the general architecture easy to understand so it is easy to modify the program, and because programs to which user information was added during transformation could pose a security problem.

5.4. **Installing to e-learning systems.** Interpreter CGI makes it possible for a person to create a function they need and then use it. Also, it enables release and sharing in class of functions that have been created by others. Some of the functions that learners and teachers create have a higher frequency of usage. For those functions to be used, the functions need to be released and, even if released, the functions need to be shared. Also, that specification program must be selected through interpreter CGI. So, a mechanism that will allow for the incorporation of these frequently used specification programs, as created by users, into an e-learning system in the same manner as regular CGI, and then allow for sharing of it with all other users, must be considered.

For this, it is necessary to compile (transform) the specification program. Through a compile system, the user-created specification is compiled as a web application and can be provided to other users as regular CGI. This is carried out by the system developer.

Conventionally, a system developer designed and incorporated e-learning system functions from scratch. However, with user-created specifications, functions are automatically created, so development costs can be reduced, and voluntarily sharing and allowing selection of the specification program is no longer needed. Also, the interpreter CGI executes the following action every time it launches.

- Transformation of specification $\Longrightarrow$ Execution of loading on the operating environment

Because of this, execution takes time, but function that were created by the compile system as regular CGI could reduce their execution time.

5.5. **Advances in e-learning systems.** In this system, desired functions are created using specification programs and, by registering them with registration CGI, these functions can actually be utilized. Interpreter CGI is used for utilization and, through release and sharing of the specification program, an environment which allows the re-use of created functions is made possible. Also, by compiling frequently used specification programs with the compile system, these functions can be provided to learners and teachers as regular CGI. Therefore, in addition to system developers, learners and teachers can create functions with specification programs which results in the advancement of e-learning system functions, and re-use of functions through release and sharing will result in the growth of e-learning system functions.

6. **Experiment.** To verify the effectiveness of specification programs, experiments were conducted to compare an existing CGI program and a specification program under conditions as follows.

Experiment 1: Example of creating table and bulleted list
Output a 3×3 table and a 3-row bulleted list under the table.

Experiment 2: Example of creating database access
Access already-solved web practice problem data and output the results in a table.

Experiment 3: Example of creating functions that can actually be used in class
A teacher checks over assignments submitted by all students, and students with high attendance rates are output in a table by descending order.

Experimental results and comparison. The experimental results are shown in the following table.

TABLE 3. CGI program and specification program comparison

Subject compared	Experiment No.	CGI program	Specification program	Ratio (Specification/CGI)
Rows	1	43 rows	13 rows	0.30
	2	168 rows	52 rows	0.31
	3	260 rows	121 rows	0.47
Bytes	1	1.33KB	0.41KB	0.31
	2	5.39KB	1.67KB	0.31
	3	8.80KB	4.40KB	0.50

Cases where directly described by a CGI program and cases where described by a specification program are compared. With regards to the resultant source code, the amount is shown in Table 3. For any of the created examples from Experiment 1 through Experiment 3, the specification program's described amount of code was less than half of those from the CGI program. Because web application elements are described in an abstract manner by the specification program, the amount of code naturally decreases; however, when looking at the code, it can be said that the transitions relationship to client from server is made clearer by the specification descriptions.

7. Effectiveness and Application of the Proposed System.

7.1. Effectiveness of the proposed system. In the following, an example is used to explain the proposed system's effectiveness.

Suppose there are two teachers, A and B, who use the proposed system in the same time period. Teacher A uses the proposed system for a certification course while Teacher B uses it for an employment support program. Teacher A can create functions that enable course material browsing and provide four-choice multiple questions to check each student's level of understanding. Teacher B can create functions that show a list of hiring companies, sort companies by business type and output a list of companies that are similar businesses to the company entered in the search box. As demonstrated here, under the proposed system, each teacher can independently construct, and utilize, functions within the same system.

Also, unlike conventional systems (see Section 2.4.1), by allowing teachers to construct functions by themselves, these functions better reflect each teacher's specific demands. Along with teachers, learners can also easily create functions through the use of a single language and those created functions can be shared with any system users.

7.2. Application of the proposed system. The proposed system can be applied to various fields including lecture/seminar education, programming, certificate acquisition and employment support.

Also, because the proposed system is created based on the ET model theory and ET program framework, it is possible to construct high-level functions as described below.

1. Function to encourage play
   A major challenge with e-learning systems is a decrease in motivation among learners. One solution to overcome the challenge can be the use of social network puzzle games. For example, a clue to the puzzle may be given each time a learner solves a course material question.
2. Interactive communication function for teacher-learner, learner-learner or learner-system
   When a learner has a question but a teacher is not immediately available, the system or other learners can provide interactive responses.
3. Learning-enhancing multiple-choice questions
   Difficult-to-answer multiple-choice questions will be created to enhance understanding levels for learners. That is, the learner must develop a higher level of understanding to select the correct answer because these multiple choices have very similar content.
4. Automatic incorrect rule detection in programming exercises [18]
   When a learner submits a program, it is marked automatically with the system detecting incorrect rule(s) and then returning the results to the learner.

8. Conclusions.

This study examined the method for easily creating system functions and proposed an e-learning system which will allow learners and teachers to create the unique functions they need.

Specifically, we proposed and incorporated the following:

- Specification programs that model e-learning system functions
- Systems that run a specification program as a web application

A specification program is one that models CGI programs, HTML and database access – the elements of a web application – in an abstract manner, so through the use of the specification program, e-learning system functions can be expressed in a single model.

While only system administrators can create and incorporate functions into conventional e-learning systems, within the framework of specification programs and the systems proposed in this study, users can easily create needed functions, easily run those functions and are able to add those functions to an e-learning system. From this, we believe with the creation of functions by users, e-learning system functions will expand and grow.

REFERENCES

[1] ALIC, *E-learning White Paper 2003/2004*, Ohmsha, Ltd., 2003.

[2] K. Ito, What's e-learning, *Information Processing*, vol.43, no.4, pp.394-400, 2002.

[3] K. Nakabayashi, Core technologies for e-learning and its standardization, *Information Processing*, vol.43, no.4, pp.401-406, 2002.

[4] K. Akama, H. Koike and H. Mabuchi, A theoretical foundation of program synthesis by equivalent transformation, *Perspectives of System Informatics, Lecture Notes in Computer Science*, vol.2244, pp.131-139, 2001.

[5] K. Akama, E. Nantajeewarawat and H. Koike, Program synthesis based on the equivalent transformation computation model, *Proc. of the 12th International Workshop on Logic Based Program Development and Transformation*, Madrid, Spain, pp.285-304, 2002.

[6] Y. Mori, *ActionScript+CGI Programming*, Softbank Publishing, 2002.

[7] R. Sugawara and E. Muraki, Theoretical re-examination of instruction design for today's e-learning, *Educational Informatics Research*, no.2, pp.131-142, 2004.

[8] K. Sano, Y. Takatori, S. Chou, T. Sasaki and H. Kobayashi, Effects of inserting answers into test sentences on students' learning achievements due to the enhancement of efficiency in answering, *Educational Technology Research*, vol.32, no.1-2, pp.91-100, 2009.

[9] H. Kobayashi, The practice of study on heredity by use of teaching materials for web based training system in a correspondence high school, *Educational Information Research, Japan Society of Educational Information*, vol.23, no.4, pp.27-34, 2008.

[10] I. Takarajima, K. Akama, I. Imani and H. Mabuchi, Design of nondeterministic program termination based on the equivalent transformation computation model, *J. of Advanced Computational Intelligence and Intelligent Informatics*, vol.10, no.3, pp.349-361, 2006.

[11] W. W. Lee and D. L. Owens, Multimedia-based instructional design: Computer-based training, web-based training, *Distance Broadcast Training, Performance-Based Solutions*, John Wiley & Sons, 2004.

[12] N. Cerpa, P. Chandler and J. Sweller, Some conditions under which integrated computer-based training software can facilitate learning, *Journal of Educational Computing Research*, vol.15, no.4, pp.345-367, 1996.

[13] A. Barron, Designing web-based training, *British Journal of Educational Technology*, vol.29, no.4, pp.355-370, 1998.

[14] B. Offir and J. Lev, Constructing an aid for evaluating teacher-learner interaction in distance learning, *Educational Media International*, vol.37, no.2, pp.91-97, 2000.

[15] K. Akama and E. Nantajeewarawat, Formalization of the equivalent transformation computation model, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol.10, no.3, pp.245-259, 2006.

[16] H. Mabuchi, K. Akama and T. Wakatsuki, Equivalent transformation rules as components of programs, *International Journal of Innovative Computing, Information and Control*, vol.3, no.3, pp.685-696, 2007.

[17] K. Miura, K. Akama, H. Mabuchi and H. Koike, Theoretical basis for making equivalent transformation rules from logical equivalences for program synthesis, *International Journal of Innovative Computing, Information and Control*, vol.9, no.6, pp.2635-2650, 2013.

[18] S. Miyajima, K. Akama, H. Mabuchi and Y. Wakamatsu, Automatic detection of incorrect rules in equivalent transformation programs, *International Journal of Innovative Computing, Information and Control*, vol.5, no.8, pp.2203-2218, 2009.

[19] *Moodle.org*, https://moodle.org.

[20] H. Hideto, A blended learning environment using Moodle, *Journal of Japan Society for Educational Technology*, vol.20, pp.241-242, 2004.

[21] H. Koike, K. Akama and H. Mabuchi, A programming language interpreter system based on equivalent transformation, *Proc. of the 9th IEEE International Conference on Intelligent Engineering Systems*, pp.283-288, 2005.