Journal of Theoretical and Applied Information Technology
31st October 2017. Vol.95. No 20

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ONTOLOGY-BASED MULTI-STAGE ADAPTATION TO IMPROVE INNOVATIVE FORM DESIGN

[1]U. UNGKAWA, [2]D. H. WIDYANTORO & [3]B. HENDRADJAYA
School of Electrical Engineering and Informatics
Institut Teknologi Bandung (ITB)
Bandung, Indonesia
E-mail: [1]ungkawa@gmail.com, [2]dwi@stei.itb.ac.id, [3]bayu@stei.itb.ac.id

ABSTRACT

The reuse of design artifacts is a very important paradigm since it can more efficiently produce new high quality designs instead of designing from the scratch. This paper discusses ontology-based adaptation applied in the XReformer system, which implements the reuse of HTML form design with a case-based design (CBD) approach. In a CBD system, adaptation plays a very important role and becomes the most complicated stage. With a design-grammar and ontology-based approach, this adaptation is expected to improve an innovative design or even become a stepping-stone toward a creative design. In this paper, we propose a multi-stage adaptation method consisting of composition, grouping, ordering and laying out stages. The composition stage combines the required elements and the grouping stage performed in order to assure the coherence of a group of elements. The aim of ordering stage is to ensure that any element is in a proper position relative to another element. The last stage arranges the elements in a right position physically. The evaluation results have shown that this adaptation is able to reuse and generate new (innovative) form designs.

Keywords—*Form-Based User Interface, Design Reuse, Form Ontology, Case-Based Reasoning.*

1 INTRODUCTION

The reuse of well-tested designs plays a key role in decreasing design times, increasing design quality, and improving the predictability of designs [1]. The reuse of design artifact can be more efficient. Therefore, in our work, we have studied the reuse of HTML form design with a case-based design (CBD) approach [2]. In our previous work, we have implemented the concept and called XReformer system. In the XReformer system, an adaptation method is a core part of the system to adapt a previous form design to meet a new requirement.

A design is a cognitive activity, and creativity is the backbone of competence desired by both individuals and organizations [3]. The design is divided into two categories: routine and non-routine. Innovative and creative designs are categorized as the non-routine design [4]. Coyne views design as a search within the design space. Innovative design is characterized by the process of exploring the design in a given space. While the creative design, including exploration outside a certain space [5]. The design is a creative process that requires reasoning, skill, and experience.

In design, one's experience plays a major role in solving new design problems. In dealing with design issues, people remember past experiences and then adapt them according to the design problem they are facing. Thus, the reasoning method using experience, i.e. case-based reasoning (CBR) would be suitable for use in design [6]. In addition, there is a suitability between the design task and the CBR process [6].

CBR is preferred over rule-base reasoning (RBR) because firstly, as an element of a knowledge base, cases are easier to acquire than rules, so case bases are more easily formulated [7], [8]. Secondly, RBR is difficult to maintain because there is a ripple effect, changing one rule will affect other rules. Renewing one case will not affect the other [7]. Thirdly, the RBR demands new problems similar to the old ones, to get the existing solution while CBR can accept partial matching problems, whereas two really equal problems are hard to encounter. In other words, CBR is more flexible than RBR [8].

Despite its advantages, CBR also introduces new difficulties. The easiness in knowledge acquisition over rule-based systems is just one type of knowledge a CBR must have. There is some

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

other knowledge required in CBR: knowledge for retrieval, domain description, adaptation and maintenance [7]. This stage of adaptation is seen as the most difficult stage [9], [10] because it is highly dependent on its domain. Patterson states that adaptation is a least-studied process [7].

Adaptation is one way of problem-solving method. The principle of adaptation is a similar problem most likely to have a similar solution. The advantage is that the problem-solving process from existing ones will be faster than finding a solution from scratch. However, an adaptation which is the most difficult task in CBR, is highly dependent on the domain, making it difficult to obtain and formulate. In analytical work, such as decision support, classification, and diagnosis, this adaptation is ignored, but in synthetic tasks such as design, configuration, and planning, this adaptation is inevitable [11].

The problem here is how an adaptation can improve and aid the user to produce the innovative HTML form design? In this paper we proposed a new adaptation method to support a new (innovative) form design. The adaptation uses a case base and a form ontology as a source of adaptation knowledge. The adaptation consists of four stages: composition, elements grouping, ordering and layouting. The case base is used to define sample forms specification and their design. The ontology is used as a form field dictionary along with its relation so that it can produce an unlimited number of forms unless there any constraint that must be met. It means that the adaptation can improve the generation of the innovative form design.

2 DESIGN, CREATIVITY, AND ADAPTATION

In this section, the relationship between design, innovation, creativity, and adaptation is presented briefly. To achieve innovative or even creative design, in CBR there must be adaptation process and other paraphernalia.

2.1 Design

Gero classifies designs into routine and non-routine [4], [12]. Then the non-routine design is divided into innovative and creative [13]. A routine design is defined as a design that takes place in a well-defined state space for potential design. Innovative designs are also in well-defined state space but the resulting design is outside the regular design space. While the creative design allows the new design to be outside of possible designs [4].

Ouyang [14] divides designs into a routine, non-routine and creative. According to him, the routine design is a design process that follows the existing scheme and the next design is inseparable from the existing scheme. The creative design is defined as a design process that goes beyond the real limits of the existing solution, and the derivative solution has a different topology than the previous design. While the non-routine design is a superset of design routine.

Creative design is preferred to produce creative products that are characterized by novelty and usability [15], [16]. To support the creative design, an approach with case-based reasoning (CBR) [17], analogical reasoning and mutation [12] are used. This is because creativity generally arises through the use of old designs in new ways [17].

2.2 Adaptation

The process in the CBR that plays an important role in the creativity of design is an adaptation. Basically, adaptations fall into 2 categories: a transformational adaptation that reuses a solution, and derivational adaptation that reuses the methods that make up the solution [18]. Derivational reuse [19] is also called generative reuse [20] which traces the derivation of a solution (problem-solving process) in accordance with the context. Wilke and Bregmann mention there is a relationship between the complexity of problem-solving and the complexity of adaptation [11]. According to them, there are five adaptation approaches from the view point of various complexity: null, transformational, generative, compositional and hierarchical adaptation. Null adaptation just takes the most similar cases then use the solution without adaptation. Transformational adaptation takes the solution of a similar old case then the solution is transformed into new solution e.g. reorganization of solution elements. Transformation also allows modification, addition, and subtraction of elements. Generative adaptation requires generating solutions from scratch that are integrated into the CBR system. Compositional adaptation combines several solutions from a number of cases. Components in the new solution consist of components from a number of solutions. In the hierarchical adaptation, cases are stored at several levels of abstraction. The solution is adapted from the highest level then step down to the lower level to get a more detailed solution.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2.3 Creative Design Characteristic

Kolodner and Will mention some of the characteristics of the creative design, first, transforming incomplete and under-constrained design requirements into more concrete and constrained ones. Second, it includes the generation process and considers some alternatives, weighing its advantages and disadvantages, and sometimes involving several parts to another. This involves utilizing the design part in a new way or modification in an unusual way [17].

Coyne explains that innovation is characterized by the exploration process within the design space. Coyne illustrates that the generation of sentences in a formal language is innovative. Generation is seen as creative if it creates a new grammar [5].

Gero distinguishes routine and non-routine design. The routine design is a simple design with only a few changes from the existing design. While the creative design is a design that is clearly different from the existing design [21]. Gero further explains the innovative design as a design activity that occurs when the context limiting the range of design variables is removed so as to allow unexpected value. While the creative design is a design activity that occurs when one or more new variables are introduced into the design [13], [21].

What is interesting is Watson and Perera's statement that creativity is more social than technical because this is the way people judge whether a design is creative or not. Creativity in design is the community's interpretation of the success and novelty of a design [6].

3 RELATED WORK

Several adaptation methods are introduced e.g. Patterson et al. used regression method in adaptation [7]. Fuchs et al. developed a differential adaptation for numerical problem-solving adaptations [22]. Gonzalez-Calero proposed a substitution-based adaptation mechanism guided by dependency relations [23]. Constructive adaptation constructs solutions through the generation of hypotheses and the preparation of hypotheses [20]. Arshadi and Badie adapted the tutoring system by composition [24].

The example of adaptation using ontology is ACook [25] which is a program for spice adaptation on recipes that consists of 3 versions of implementation: adaptation with ontology (Onto-ACook), with CBR (CBR-ACook) and with Knowledge Discovery (AKD-ACook). The adaptation uses operators: *add* and *remove* spices on a recipe. Seasonings are divided into animal origins, herbs, base spices, beverages, and sweeteners. Onto-ACook can only do simple substitution, replace spices with another, regardless of how to get a better recipe.

Another example of an ontology adaptation is ColibriCook [26], a CBR system for retrieval and ontology-based adaptation. The ontology consists of ingredients, formal type, cuisine type, dietary type and ingredient type. The ingredients have an identifier property, father-similarity, *Is-Ingredient-Type, Is-Made-Of, Availability, Is-Substitutable*.

4 FORM REPRESENTATION

A form design can be modeled as a formal grammar defined as the tuples: *Gm = (N, T, P, F, Ord, Lo)*. A grammar *Gm* consists of the following:

- *N*: A finite set of non-terminal symbols. In form design, *N* represents an element grouping, sub form, and composite element.
- *T*: A finite set of terminal symbols, which is representing atomic form elements or atomic form fields.
- *P*: A finite set of production rules. A (set of) production rules can be used to represent or generate a form.
- *F*: A start symbol, here, it means a form.
- *Ord*: fields ordering is a subset of a cross product of $R \to T \times O \times T$; where: $O = \{\prec, \succ\}$, $\prec$ = *before* or *precedes*, and $\succ$ = *after* or *succeeds.*
- *Lo*: lay out:is a subset of a cross product of $R \to T \times Lot \times T$; where : $Lot = \{\nabla, \Delta, \triangleleft, \triangleright\}$, where ∇ = above, Δ = below, $\triangleleft$ = left, $\triangleright$ right.

The non-terminal symbols always begin with a capital letter and the terminal symbols begin with a small letter. The examples of non-terminal symbols *N* are *field group* (*G*), *input fields (I)*, etc. The terminal symbols *T* examples are *first_name (fname), retype password (rpwd)*, etc. To describe form generation using formal grammars, the first step is to define the set of production (rewriting) rules of the grammar, which will drive the generative process. The production rules *P* below describes the grammar (language) of forms:

$$F \to G \mid G\ G$$
$$G \to G\ G$$
$$G \to I \mid O \mid C \mid I\ I \mid I\ O \mid I\ C \mid O\ O \mid O\ C \mid C\ C$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$I \rightarrow Id \mid Name \mid Addr \mid Phn \mid Pwd$

$I \rightarrow Lang \mid Menu \mid Icap...$

$O \rightarrow Msg \mid Ocap \mid ...$

$C \rightarrow Lgn \mid Rset \mid Rcap \mid ...$

$Id \rightarrow uid \mid userid \mid ...$

$Pwd \rightarrow pwd \mid repwd$

$Lgn \rightarrow lgn$

$Lang \rightarrow lang$

$Menu \rightarrow menu$

$Rset \rightarrow rset$

$Icap \rightarrow icc$

$Ocap \rightarrow occ$

$Rcap \rightarrow rcap$

The first 3 lines rules state that form *F* consists of one or more groups (*Gr*). Here, the group may represent field group, composite field or subform and it consists of one or more sub group. The group also can be composed of input fields (*I*), output fields (*O*), control buttons (*C*), and some combinations of *I, O* and *C*.

For example, given the following form (Figure 1):

Figure 1. A sample form.

The above form consists of user id (*uid*), password (*pwd*), captcha generator (*occ*) and refresh captcha button (*rcap*), captcha input (*icc*), login, and reset buttons, language (*lang*) and menu options. The elements of the form can be divided into many groups, such as: *(uid, pwd)*, *(lang, menu)*, *(occ, rcap)*, *(icc)*, *(login* and *reset)*. To generate a form, we apply some rewriting rules as follows (Figure 2):

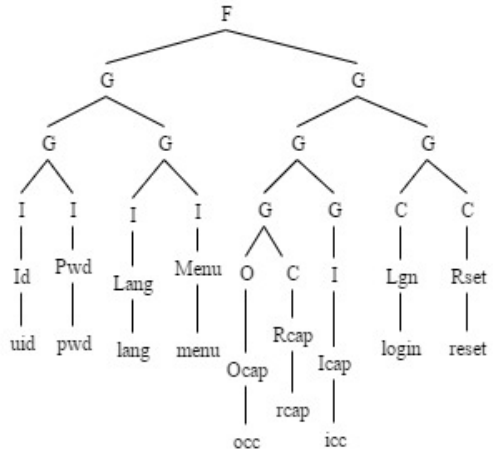

Figure 2. A derivation tree.

The fields order can be represented as: $uid \prec pwd, lang \prec menu$. They indicate that *uid* precedes *pwd* and *lang* precedes *menu*. The fields lay out can be represented as $uid \nabla pwd, occ \triangleleft rcap$. They say that *uid* is above of *pwd* and *occ* is on the left of *rcap*.

5 FORM DESIGN ADAPTATION

The adaptation of form design is the adaptation of the solution in a case (from the case base) consisting of specifications and designs of a set of forms in the XReformer system. This system is an ontology-based CBR system. So the XReformer knowledge base consists of a dynamic case base and a static ontology. The case base becomes form-specific representation and ontology becomes a generic form representation.

The adaptation of form design has to meet the requirements of the form design that has some problems (issues) as follows:

- An obligatory element: Is there a mandatory element not listed in the form (the proposed design)?
- An odd (unusual) element in a form: Is there a strange element listed in the proposed form?
- Two equivalent elements: Is the proposed form freed from the same (semantically) elements in the form?
- A substitutable element: Can the new elements be replaced by other elements?
- An odd element in a group: Are the newly added elements acceptable in the group?
- Order: Did the new element already ordered in the right position?

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- Lay out: Did the new element already laid out in the right position?
- Inconsistency. For example, the order of some elements is inconsistent.
- Incompleteness: Can the adaptation process produce a form from an incomplete specification?
- Variety: For example, The login form does not necessarily consist of a username and password, it can also be email and password.
- Innovative: The adaptation process should be able to produce new and useful designs.

Broadly speaking, the adaptation of the form design consists of several stages. The first stage is to compare the form specification of the user (normalized query) to the form specification of the best case retrieved from the case base. The query stated in the semi-formal representation. This comparison generates a number of form elements (fields) to be added and a number of elements to be deleted (composition). From a number of added elements (the specification added) are taken one by one to be inserted into the form. To be able to determine where the added element is inserted, first, the new elements should be combined to retrieved groups (combination). Each combination then is evaluated for the adjacency of all elements (grouping). The highest score combination is then promoted to become a new group. After the element is entered in the right group, then the order is determined (ordering). This step can help the next step of a layouting process.

5.1 Composition

Conceptually, the design of the form is derived from the composition of various elements in the ontology by following several cases in the case base. Generally, a CBR that does not use an ontology, the composition process only depends on the case base. The composition in the form design is limited by some constraints such as the presence of mandatory elements and/or banned (odd) elements in the form.

In the removal process of elements, an element can be removed if the element is not required in the desired form. If the element is required, then the element remains included in the proposed design. Conversely, if the element is strange, unusual, entered in a form, then the element can not be added. So the elements that can be removed are elements that are not mandatory and elements that can be added are elements that are not strange/odd. But the presence of odd elements can promote innovative or even creative design. Therefore removing these strange elements are still being considered for implementation in the current adaptation. Whether the element is compulsory or prohibited in a form, is specified in the ontology.

Determining an element as a mandatory or odd element in a form to be included in an ontology requires careful consideration. The presence of odd elements can encourage creativity but can also become a redundant element or even become a forbidden element. For example, *login* form. Generally, the *login* form must contain an element of identity and password as an entry key. However, this identity can be a *user_name* or now many design use email. Then, can *first_name* be an identity for the *login* form or as an odd element? The consideration is if *first_name* becomes an identity, it will be found a lot in common because many people have the same first name. But the identity must be unique. Therefore, a differentiator must be added to the first name to be unique. This means that the first name becomes user name. Thus it is clear that the first name becomes a strange element in the login form in the position as identity.

Another thing to consider is the factor of user identification, whether a machine or human, to avoid deception. In this case, the *login* form should get an additional new variable i.e. security factor, for example in the context of an interactive session in a web site that will involve electronic transactions. The addition of a security factor in this login form includes creative design (because it adds new variables), but if later the captcha as a value of a security variable is replaced by other means, it means that the system generates an innovative design [13], [21].

Elements that are equivalent in the new form have been prevented to appears together in the new design, in the process of normalizing query done before the adaptation process. The query normalization will recognize elements that are semantically the same, such as *forename, fore_name, firstname,* and *first_name*, will be normalized to *first_name* as a standard element. So, if there is a query containing one of these elements, it will be normalized to *first_name,* because the first_name is the default term for that concept.

New elements that can be replaced (substitutable) with other elements are some elements that can occupy the same variable in a form. If this is the case, then some form with

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

various substitution elements can be shown. For example, a *user_name* element can be replaced with an *email* in the *login* form (occupying the status as the identifier). If a query contains one of those elements, then all *login* forms containing one of those elements will be displayed provided that the user specifies the number of forms to be displayed more than one. On the other hand, if the query lists both elements, then it will display some form containing only one of those elements.

In the implementation, the composition is done through a combination of newly added elements with several groups of elements from the retrieved form:

$$FieldComb = FGroupCase \times Spec_{ADD}$$

The number (cardinality) of combinations (|*FieldComb*|) equals to the product of the number of groups of the retrieved form (|*FGroupCase*|) with the number of the added elements (|$Spec_{ADD}$|). Thus, in *FieldComb* apply:

$$fc_{ij} = fg_i \cup \{ne_j\}$$

where $fc_{ij} \in$ FieldComb, $fg_i \in$ FgroupCase, and $ne_j \in Spec_{ADD}$.

5.2 Grouping

The process of grouping a new element into its group is the group's search process for that element. The search criteria are based on the proximity of new elements in the ontology (structural adjacency) with other elements already in the group. Each combination resulted from the above compositional process is evaluated for the proximity/adjacency of the elements within the combination. The structural adjacency in ontology is calculated by cosine equation [27]:

$$sim(ne, efg) = \cos ine(ne, efg) = \frac{\left|\left(\bigcup_{c \in t(ne)} (\text{super}(c, CN))\right) \cap \left(\bigcup_{c \in t(efg)} (\text{super}(c, CN))\right)\right|}{\sqrt{\left|\bigcup_{c \in t(ne)} (\text{super}(c, CN))\right|} \cdot \sqrt{\left|\bigcup_{c \in t(efg)} (\text{super}(c, CN))\right|}}$$

where

- CN is the set of all concepts/classes in the ontology
- super (c, CN) is a sub-set of the CN that becomes super concept/class of class c.
- t (i) the set of class/concept of individual i.

The cosine equation states that the similarity of elements *ne* with elements *efg* is the number of elements (cardinality) of intersection set between a set of super-class of *e* with a set of super class of *efg*, divided by the product of the square root of the cardinality of a set of super class *e* with the square root of the cardinality of a set of super class *efg*. What is meant by "super class" here is all class (super class) of the belonging element.

In the cosine equation above, *ne* is a newly added element while *efg* is an element already present in the group. Thus, there will be as many *n* values of evaluation where $n = |fg|$. The values of this cosine equation are summed up to become a value of the average score:

$$\overline{score}(fc) = \frac{\left(\sum_{i=1}^{n} sim(ne, efg)\right)_{fc = fg \cup \{ne\}, efg \in fg, n = |fg|}}{n}$$

The above average score is the score value of each combination as a new candidate group in the form because, in fact, this combination is the result of a combination of the old group with the new elements. Thus, the new group is grabbed from the combination with the highest average value:

$$nfg = \arg\max_{fc \in FieldComb} \overline{score}(fc)$$

After determining the new candidate group from the set of combinations, then the old group should be substituted with the new one. The old group that was removed was the group that became the new successor group:

$$FGroupCase = FGroupCase \ / \{fg(id)\}$$

$$FGroupCase = FGroupCase \ \cup \{nfg(id)\}$$

Thus the grouping process must begin with a combination, and then an evaluation, selection, and substitution. This grouping process can be viewed as a semantics classification or featureless classification. It is said to be a semantic classification because elements of one group come from a family or adjacent classes. Thus, no feature is used to define a group of elements.

5.3 Ordering

The ordering process acts as an intermediary from the grouping process to lay outing process of the elements. Through this stage, the elements are arranged, sorted and determined its relative position against another element. The ordering relationships are stored in an ontology as *elementA isPredecessorOf elementB*.

The difficulty that arises in determining the relative position of elements in the form ontology

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

is inconsistency. A form may consist of *first_name*, and then last name or it could be in the reverse order: *last_name* (*family_name*) first then *first_name*. We should specify the default order *first_name* and *last_name* in accordance with the meaning of the *first* and *last* words, except in certain contexts that require the opposite. If so, we need the context that determines it.

Generally speaking, the ordering process takes place in two stages: the search for the group in which its elements to be ordered. Once the group obtained, then perform the placement of new element relative to another element in the group. In positioning, the new element is compared to each element already present in the group based on the relative positions, in the groups specified in the ontology.

5.4 Layouting

This stage requires relative position information resulting from the ordering stage. The layouting stage is required as a bridge for the generation of the real form. With layout information, the form generator determines the position of the element whether it is placed vertically or horizontally. Without layout information, the position of the element is still ambiguous, whether an element is above (vertical) or on the left (horizontal) of another element. In the real form, there is also a horizontal or vertical design. So element ordering information can also be viewed as an abstraction of layouting information.

In the implementation, the layouting process should consider whether an element is inline with another element (as row elements) or the element is aloof inside a row (as column element). If several elements are lined up in a row, it must be determined when the row begins (an element become the first element in a row) and when it ends (an element become the last element in that row), since it can be more than two elements in a row.

5.5 Innovative and Creative Form Design

To ease and simplify the evaluation of the system, here we define the innovative and creative design. First, it must be assumed that the case base has kept the form cases (specifications and design) completely so that if there is a new design that was not originally in the case base then the design is considered innovative. For example, until recently, a *login* form only recognizes a *user_name* as the identity. Then lately *email* appears instead of *user_name*. In this case, then the new design is considered innovative.

Furthermore, in order to determine the degree of creativity of the design, the form is assumed to have an abstract representation. For the case of the *login* form, for example, the form consists of identity and key as an abstraction (as a variable of design) of the form for *user_name* and *password*. If there are other variables added e.g. security with captcha (as its value) for example, then the design is considered creative. To be able to meet both of these things, it is required a complete ontology that already represents all elements of the form that may exist in the case base.

6 EVALUATION

This evaluation of adaptation performance is based on the adaptation requirements as mentioned above. The evaluation should be able to ensure that the adaptation method meets the requirements. Table 1 lists the test scenarios for each item of requirement.

Table 1. The adaptation test scenario

Requirements	Test scenario and success criteria
Mandatory element	- The NQ (normalized query) contains form name and no mandatory element. - The result shows the mandatory element.
Odd element	- The NQ contains form name and an odd element. - The result shows no odd element.
Equivalent element	- The NQ contains form name and a non-standard element. - The result lists no non-standard element and shows the semantically similar element.
Substitutable element	- The NQ contains form name and two substitutable elements, one to another. - The result only lists one of the two substitutable elements.
Grouping	- The NQ contain form name and a new element. - The new element inserted in the closest elements in a group.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Requirements	Test scenario and success criteria
Odd element in a group	- A test form only contains one group. - The NQ contains form name and an odd element in the group. - The result lists no odd element.
Ordering	- The NQ contains form name and a new element. - The result shows the new element located accordingly
Laying out	- The NQ contains form name and a new element. - The result shows the new element laid accordingly
Inconsistency	- The NQ contains form name and two elements. - The result shows the standard ordering
Incompleteness	- The NQ contains form name. - The result shows the complete form
Variety	- The NQ contains form name. - The result shows various forms
Innovativeness	- The NQ contains form name. - The result shows new forms

Table 2 below provides the results of the adaptation evaluation. The second point in the second column is the best case retrieved from the case base. It needs to be shown in order to be able to compare the query with the retrieved case.

Table 2. The evaluation results

Requirements	Test results
Mandatory element	- The NQ contains login form and first_name, has no password as a mandatory element. - The best old design: user_name, password, submit. - The new design shows password as the mandatory element.
Odd element	- The NQ contains login form and first_name as an odd element. - The best old design: user_name, password, submit. - The new design shows no first_name.
Equivalent element	- The NQ contains registration form and forename as a non-standard element. - The best case: email, password, last_name, first_name, submit. - The adapted case lists no forename but shows a first_name.
Substitutable element	- The NQ contains login form, user_name, and email as two substitutable elements, one to another. - The result only lists one of the two substitutable elements.
Grouping	- The NQ contains login form and captcha - The best-retrieved case contains user_name and password - The captcha inserted in a group of user_name and password
Odd element in a group	- Similar to the odd test in a form.
Ordering	- The NQ contains form name and a new element. - The result shows the new element located accordingly
Laying out	- The NQ contains registration form and a new element. - The best case shows email, password, last_name, first_name. - The result shows email password country last_name first_name username, in order
Inconsistency	- Similar to the ordering test
Incompleteness	- The NQ contains login form. - The best old design: user_name, password,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Requirements	Test results
	submit. - The result shows the complete form. - Or - The NQ contains user_name. - The best old design: user_name, password, submit. - The result shows the complete form.
Variety	- The NQ contains login form. - The result shows various forms if the user asks more than one. - Form1: user_name, password, submit - Form2: email, password
Innovativeness	- The NQ contains login form and user_name. - The result shows new forms: with email,

7 DISCUSSION

The proposed adaptation method still derives the new solution from the previous case. At this stage, the adaptation is still at an innovative level and not yet a creative one, since it still adapts the old solution [28].

In addition, Coyne insists that the generation of sentences in a formal language is an innovative process. A generation is seen as a creative process if it can create a new grammar [5]. In accordance with the form generation representation with formal grammar as described above, the form design included in a grammar design. Thus the form generation is not yet a creative one. However, the ontology engagement in adaptation places the form ontology as a general form representation to distinguish it from a specific case representations and encourages for better creative designs.

8 CONCLUSIONS AND FUTURE WORKS

The adaptation method that has been developed in XReformer system that we built have successfully fulfilled such requirements such as to generate new form designs, able to handle incomplete queries and more. That is, this adaptation has proven to able to produce an innovative design. However, the challenge in this adaptation is creativity. Through the experience of implementation and evaluation, we have obtained several points to be developed further, especially related to the creativity of design.

The first is a form abstraction, in the sense that the form is not represented as a set of elements directly but with an abstract representation. For example, the *login* form is not composed of *user name* and *password* but with *identity* and *key*. The identity can be a *user name* or *email* or probably another (as the value of an identity variable) element. Likewise, a key can be a *password*, *pin* or something else. Thus this abstract representation can be viewed as a variable in the form design. The addition of new variables in a form will be a creative design. Another way might be merging two or more variables into one. This can also be seen as creative designs. Secondly, the ontology should be made as complete as possible, covering all existing data of concepts (in the case base). It needs the acquisition and abstraction of data from various databases. Thirdly, can the abstract representations in forms and the adaptation be applied in other domains such as room layout, musical compositions, and recipes?

One of the open areas in this research is how the ontology based adaptation being applied in other domains especially to support creativity designs. This work clues the involvement of ontology to represent the abstract interrelationship between elements and how the grammar of the representation being transform to accommodate the computational creativities since in this case, the creativity is indicated by the ability to process the variable in the grammar.

REFERENCES

[1] K. Börner, "CBR for Design," *Case Based Reason. Technol. From Found. to Appl.*, vol. 1400, pp. 201–234, 1998.

[2] D. H. Widyantoro, U. Ungkawa, and B. Hendradjaya, "Case-based reasoning approach for form interface design," in *Proceedings of 2014 International Conference on Data and Software Engineering, ICODSE 2014*, 2014.

[3] C. S. Lee, J. L. Kolodner, and A. K. Goel, "Guest Editorial - Creative Design : Scaffolding Creative Reasoning and Meaningful Learning," *Educ. Technol. Soc.*, vol. 14, no. 1, pp. 1–2, 2011.

[4] J. S. Gero, "Design Prototypes: A Knowledge Representation Schema for Design," *AI Mag.*, vol. 11, no. 4, p. 26,

1990.
[5] R. D. Coyne, M. A. Rosenman, A. D. Radford, and J. S. Gero, "Innovation and creativity in knowledge-based CAD," *Expert Systems in Computer-Aided Design*. pp. 435–465, 1987.
[6] I. Watson and S. Perera, "Case-based design: A review and analysis of building design applications," *J. Artif. Intell. Eng. Des. Anal. Manuf. AIEDAM*, vol. 11, no. 1, pp. 59–87, 1997.
[7] D. Patterson, N. Rooney, and M. Galushka, "A Regression Based Adaptation Strategy for Case-Based Reasoning," *AAAI*, pp. 87–92, 2002.
[8] R. Mitra and J. Basak, "Methods of case adaptation: A survey," *Int. J. Intell. Syst.*, vol. 20, no. 6, pp. 627–645, 2005.
[9] Y. Zhang, P. Louvieris, and M. Petrou, "Case-Based Reasoning Adaptation for High Dimensional Solution Space," *Trans. Case-Based Reason. Multimed. Data*, vol. 1, no. 1, pp. 21–36, 2008.
[10] D. B. Leake, A. Kinley, and D. Wilson, "Acquiring Case Adaptation Knowledge : A Hybrid Approach," in *13th National Conference on Artificial Intelligence*, 1996.
[11] W. Wilke and R. Bergmann, "Techniques and Knowledge used for Adaptation during Case-Based Problem Solving," 1997.
[12] J. S. Gero and M. Lou Maher, "Mutation and analogy to support creativity in computer-aided design," *CAAD Futur. '91*, 1991.
[13] J. S. Gero, "Computational Models of Creative Design Processes," *Artif. Intell. Creat.*, 1994.
[14] J. Ouyang, "Case-based Reasoning for the Creative Design of Electromagnetic Devices," no. November, p. 128, 2011.
[15] P. Gomes, F. C. Pereira, C. Bento, and J. L. Ferriera, "Using Analogical Reasoning to Promote Creativity in Software Reuse," pp. 152–158, 2001.
[16] R. Saunders, "Curious Design Agents and Artificial Creativity," 2002.
[17] J. L. Kolodner and L. M. Wills, "Case-based Creative Design," 1993.
[18] A. Aamodt and E. Plaza, "Case-Based Reasoning : Foundational Issues , Methodological Variations , and System Approaches," *AI Commun.*, vol. 7, pp. 39–59, 1994.
[19] J. G. Carbonell, "Derivational analogy : a theory of reconstructive problem solving and expertise acquisition," 1985.
[20] E. Plaza and J.-L. Arcos, "Constructive Adaptation," in *Advances in Case-Based Reasoning 6th European Conference, ECCBR 2002 Aberdeen, Scotland, UK, September 4-7, 2002 Proceedings*, 2002.
[21] J. S. Gero, "Computational Models of Innovative and Creative Design Processes," *Technol. Forecast. Soc. Change*, vol. 64, no. 2–3, pp. 183–196, 2000.
[22] B. Fuchs, J. Lieber, A. Mille, and A. Napoli, "Differential adaptation: An operational approach to adaptation for solving numerical problems with CBR," *Knowledge-Based Syst.*, vol. 68, pp. 103–114, 2014.
[23] P. A. González-Calero, M. Gómez-Albarrán, and B. Díaz-Agudo, "A Substitution-based Adaptation Model," in *Challenges for Case-Based Reasoning - Proceedings of the ICCBR'99 Workshops*, 1999, pp. 17–26.
[24] N. Arshadi and K. Badie, "A compositional approach to solution adaptation in case-based reasoning and its application to Tutoring Library," *Proc. 8th Ger. Work. Case- …*, 2000.
[25] S. G. Mota and B. D. Agudo, "ACook: Recipe adaptation using ontologies, case-based reasoning systems and knowledge discovery," *Comput. Work.*, p. 41, 2012.
[26] J. DeMiguel, L. Plaza, and B. Dıaz-Agudo, "ColibriCook: A CBR System for Ontology-Based Recipe Retrieval and Adaptation," *9th Eur. Conf. Case-Based Reason. - ECCBR*, pp. 199–208, 2008.
[27] J. a Recio-García, B. Díaz-Agudo, and P. González-Calero, *jCOLIBRI2 Tutorial*. 2008.
[28] S. Bailey and I. Smith, "Case-Based Preliminary Building Design," *J. Comput. Civ. Eng.*, vol. 8, no. 4, pp. 454–468, 1994.