

This is an Open Access document downloaded from ORCA, Cardiff University's institutional repository:<https://orca.cardiff.ac.uk/id/eprint/68109/>

This is the author's version of a work that was submitted to / accepted for publication.

Citation for final published version:

Li, Luye, Qin, Feiwei, Gao, Shuming and Liu, Ying 2014. An approach for design rationale retrieval using ontology-aided indexing. *Journal of Engineering Design* 25 (7-9) , pp. 259-279.  
10.1080/09544828.2014.969690

Publishers page: <http://dx.doi.org/10.1080/09544828.2014.969690>

Please note:

Changes made as a result of publishing processes such as copy-editing, formatting and page numbers may not be reflected in this version. For the definitive version of this publication, please refer to the published source. You are advised to consult the publisher's version if you wish to cite this paper.

This version is being made available in accordance with publisher policies. See <http://orca.cf.ac.uk/policies.html> for usage policies. Copyright and moral rights for publications made available in ORCA are retained by the copyright holders.



## RESEARCH ARTICLE

### An approach for design rationale retrieval using ontology-aided indexing

Luye Li<sup>a</sup>, Feiwei Qin<sup>a</sup>, Shuming Gao<sup>a\*</sup> and Ying Liu<sup>b</sup>

<sup>a</sup>*State Key Lab. of CAD & CG, Zhejiang University, Hangzhou, China;* <sup>b</sup>*Institute of Mechanical and Manufacturing Engineering, School of Engineering, Cardiff University, Cardiff, UK*

*(Received 00 Month 200x; final version received 00 Month 200x)*

Design rationale is an important category of design knowledge. Effective reuse of design rationale depends on its successful retrieval. In this paper, an approach for design rationale retrieval using ontology-aided indexing is presented. First, a design rationale ontology is designed based on the extended IBIS-based design rationale representation in order to effectively utilize the semantics embedded in DR. Then, an ontology-aided indexing method is proposed to build indexes for design rationale records to index the semantic concepts and relationships in DR. Furthermore, three kinds of query modes are developed to support flexible querying, among which natural language input query and DR record based query have much more semantics and thus lead to better retrieval results. Finally, a prototype system is implemented. The experimental results show the effectiveness of the proposed approach.

**Keywords:** design rationale retrieval; design rationale; ontology; design knowledge reuse

## 1. Introduction

In product design and development, design engineers carry out various activities related to analyzing requirements, proposing and evaluating solutions, and making decisions. To complete these tasks, knowledge reuse of previously proven designs is indispensable. Design rationale (DR) contains most of this kind of design knowledge and know-how because it is centric to any design activity and process and it includes all the background knowledge such as design deliberation, reasoning, trade-off, and decision-making in the entire design process of an artifact - information that can be valuable, even critical, to various people who deal with the artifact (Regli *et al.* 2000). In recent years, more and

---

\*Corresponding author. Email: smgao@cad.zju.edu.cn

more companies have become aware of the importance of DR capture and reuse. As pointed out in (Kim *et al.* 2005), knowledge reuse involves three activities: (1) searching for similar problems or design cases; (2) recognizing reusable parts of knowledge; and (3) adapting the retrieved knowledge to new requirements. Therefore, an effective reuse of DR highly depends on the successful retrieval of relevant DR information.

Research concerning how to capture, store, and retrieve DR has been consistently undertaken in the past 40 years, and several DR systems have been developed. However, most of them cannot take full advantage of the semantics embedded in DR so that the retrieved results are far from desired. Although there are a few tools which represent DR using ontology, they are not easy to be used by the end-users because retrieving DR with rich semantics needs formal query languages and formulating a query using such languages normally requires the knowledge of domain ontology as well as the syntax of the language (Kara *et al.* 2012).

In this paper, a DR retrieval approach using ontology-aided indexing is presented which aims to tackle the aforementioned problems. First, in order to take advantage of the semantics embedded in DR, a DR ontology is designed according to the proposed extended IBIS-based DR representation. Based on the DR ontology, the captured DR records are transferred to ontology individuals through ontology population. Then, the ontology information in DR database is enriched by semantic rule-based reasoning, and the DR database is indexed with the aid of the ontology information to improve the performance of DR retrieval. Finally, three kinds of query modes are provided to the end-users in relation to different knowledge requirements. Among such query modes, natural language input query and DR record based query which contain more semantic information yield higher performance in terms of retrieval recall and precision.

The rest of this paper is organized as follows: Section 2 presents the state of the art of DR representation and retrieval. Section 3 shows the overview of our approach. Section 4 details the design of a DR ontology based on the extended IBIS-based DR representation. Our proposed approach of ontology-aided indexing for DR search and retrieval is reported in Section 5. Three user-friendly query modes for DR querying and query processing are given in Section 6. After that, the implementation of our prototype system is described in Section 7. Finally, Section 8 summarises the proposed work.

## 2. Related work

### 2.1. DR representation

A good representation schema is vital to enabling effective design and reuse (Regli *et al.* 2000). Research on DR representation has been reported since the 1970s. Most of the DR representation approaches are argumentation-based approaches, and the typical model is issue-based information system (IBIS) (Kunz and Rittel 1970), which uses issues, positions, arguments and relationships between them to represent DR. Several software tools which allow engineering designers to record DR have been implemented based on IBIS. For example, Conklin and Begeman (1988) developed graphical IBIS (gIBIS), and Bracewell *et al.* (2009) implemented Design Rationale editor (DRed). In addition, McCall (1991) proposed the Procedural Hierarchy of Issues (PHI) model, which broadens the scope of the concept of issue in IBIS. Another argumentation-based model is question, option and criteria (QOC) (MacLean *et al.* 1991), which is a kind of semi-formal notation of design space analysis. Liu *et al.* (2010) proposed an issue, solution and artifact layer (ISAL) model for DR representation and rationale information discovery from design

archival documents (see also [Liang et al. 2012](#)).

In addition, as the development of semantic web technology, several ontology-based representation schemas for DR information are proposed. [Burge and Brown \(2008\)](#) developed a system, Software Engineering Using RAtionale (SEURAT) system, which extends decision representation language (DRL) with argument ontology. This argument ontology is a hierarchy of common arguments that serve as types of claims. [De Medeiros and Schwabe \(2008\)](#) proposed the Kuaba Ontology, which extends the argumentation structure of IBIS explicating the representation of the decisions made during design and their justifications, and the relations between the argumentation and generated artifacts. Also based on the IBIS model, [Zhang et al. \(2013\)](#) proposed an ontology-based semantic representation model for DR information, namely the integrated issue, solution, artifact and argument (ISAA) model, which introduces the ontology-based semantic representation mode to the DR representation mode to the DR representation and expands the concept elements of IBIS. To facilitate decision making within collaborative design, [Rockwell et al. \(2009\)](#) developed a Decision Support Ontology (DSO) which includes decision-related information such as the design issue, alternatives, evaluation, criteria and preferences. It also includes decision rationale and assumptions, as well as some constraints created by the decision and the decision outcome. Although DSO which includes more element types and relations can describe DR in more explicit manner, it is not practical to capture all these DR contents.

## 2.2. DR retrieval

There are several works dedicated to DR retrieval in recent years. In general, DR retrieval works can be classified into two main categories: text-based retrieval and ontology-based retrieval.

Most of current DR retrieval methods are text-based. [Liang et al. \(2010\)](#) proposed a DR search and retrieval system which focuses on interactive user interface design. There are three basic functions: the view functions enable engineering designers to intuitively navigate DR repository; the search functions support designers to retrieve relevant DR from multiple aspects; and the analysis functions suggest some useful DR insights. Kim et al. presented two methods for the retrieval of DR captured using DRed. The first approach uses natural language processing (NLP) techniques to annotate rationale records with 9 selected semantic relations ([2005](#)). The second approach recommends relevant pieces of DR by analyzing the design task models of design reuse ([2007](#)). Also for DRed files, Wang et al. developed a keyword-based retrieval tool at first ([2009](#)), and then proposed a new DR retrieval system making use of the implicit structures in DRed graphs ([2012](#)). The general problem about the text-based retrieval is that various DR records have semantics such as types, relationships and structures, etc., however, text-based retrieval is very hard to take full advantage of the semantics.

In comparison with text-based retrieval, ontology-based retrieval makes better use of the semantics embedded in DR records by utilizing ontology. Lim et al. ([2010](#)), ([2011](#)) proposed an information search and retrieval framework based on the semantically annotated multi-facet product family ontology, and exemplified how they can derive new product variants based on the designer's query of requirements via the faceted search and retrieval of product family information. [López et al. \(2008\)](#) presented NDR ontology to describe non-functional requirements (NFR) and DR knowledge, and multi-facet search

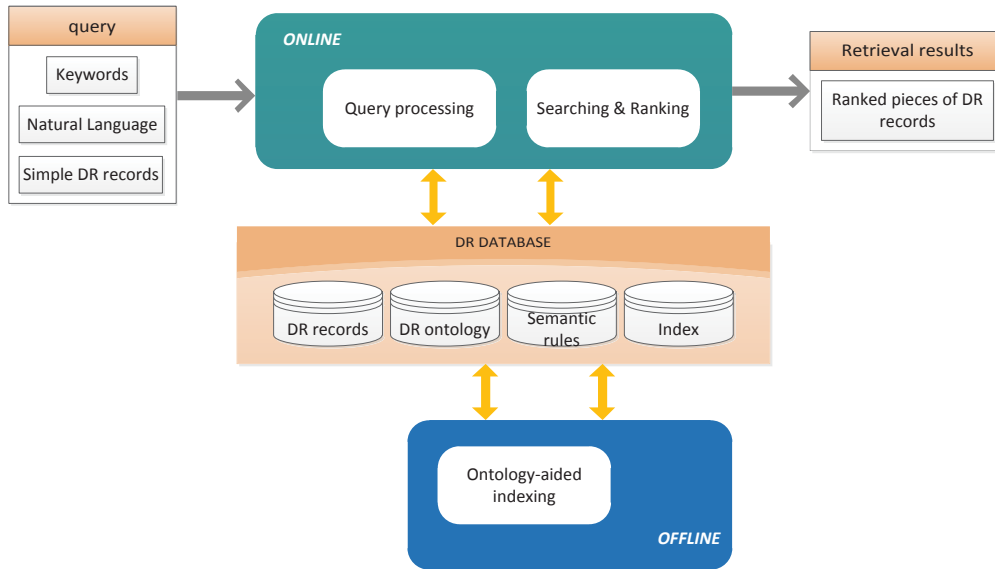


Figure 1. DR retrieval framework.

was implemented through executing SPARQL<sup>1</sup> queries over the semantic catalogues of NFR. However, these approaches are far from being practical since they require a relatively complex query language. A scalable alternative to query construction from simple queries is semantic indexing, in which semantic data in RDF<sup>2</sup> knowledge bases is indexed in a structured way and directly available to be searched with simple queries such as keyword-based query. In information retrieval domain, Kara *et al.* (2012) presented an ontology-based information extraction and retrieval system in the soccer domain, in their work, a keyword-based retrieval approach using semantic indexing was proposed.

### 3. Overview of approach

Based on the analysis of the requirements on DR retrieval for knowledge reuse, we propose a DR retrieval approach using ontology-aided indexing. The goal of the approach is to take full advantage of the semantics embedded in DR, and thus enhance the retrieval effect.

Figure 1 shows the overview of our ontology-based DR retrieval approach. It could be seen that our approach contains three main parts, i.e. the DR database, the online processing and the offline processing. Here we give a brief description of each part respectively.

The DR database stores all the necessary data involved in both online processing and offline processing, including the DR records captured by designers, the DR ontology designed according to the extended IBIS-based DR representation as well as the semantic rules defined. Moreover, the index generated by ontology-aided indexing is also stored in this database. In this work, each DR record is stored as a file whose content is represented in a structured way, similar to that in the proposed DR representation in Section 4.1.

<sup>1</sup><http://www.w3.org/TR/rdf-sparql-query/>.

<sup>2</sup><http://www.w3.org/TR/PR-rdf-syntax/>.

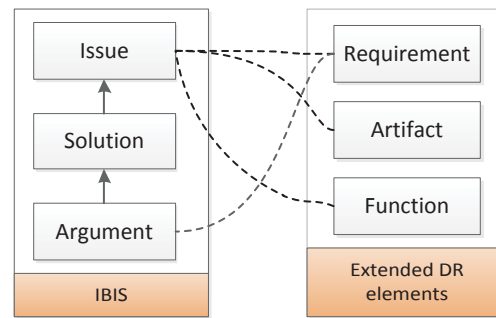


Figure 2. Extended IBIS-based DR representation.

The online processing starts when a user inputs a query and ends with exporting pieces of DR records fulfilling the query. In order to satisfy users' different requirements, three kinds of query modes are provided. For each query mode, the query is processed with a specific method firstly. Then, the processed query is taken to search the required DR records. Finally, the retrieval results are ranked in a reasonable order.

The major task of offline processing is to set up the indices for all the DR records in the DR database. In order to effectively index all the DR records, an ontology-aided indexing method is developed. The method consists of parsing of DR records, ontology population, ontology reasoning and making index for instance ontologies. Although all of these operations cost much time, it does not affect the real-time search of DR since they are pre-processing.

## 4. Ontology design

An ontology formally represents knowledge as a set of concepts within a domain, and the relationships between pairs of concepts. In order to effectively make use of the semantics embedded in DR, a corresponding ontology which contains the domain knowledge of DR can be designed to help representing DR. In this work, we develop a DR ontology based on an extended IBIS-based DR representation for DR retrieval. Before describing our ontology, the extended IBIS-based DR representation is briefly introduced firstly.

### 4.1. DR representation for knowledge retrieval and reuse

In order to effectively support the retrieval and reuse of design knowledge, a DR representation should have the following characteristics as far as possible: (1) expressive enough to represent the design knowledge generated in design process; (2) formal enough to support computation; (3) easy to be captured (Qin *et al.* 2012). However, existing DR representations are not good enough in these aspects. On the one hand, traditional DR representations do not have enough expressing ability; on the other hand, most of them are not formal enough to be understood by the computer.

Ahmed and Wallace (2004) did a comprehensive analysis of the discourse between novice designers and experienced designers and identified eleven main kinds of knowledge needs including *how does it work*, *why*, *what issues to consider*, *when to consider issues* and *design process*. IBIS (Kunz and Rittel 1970) is a traditional DR representation which can express most of the first three knowledge needs, moreover, we find that **Requirement** can answer *why* and *when*, **Function** describes *how*, and **Artifact** is highly related to



Figure 3. Class hierarchy of DR ontology.

*design process.*

Based on the analysis above, we propose an extended IBIS-based DR representation. As shown in Figure 2, the extended DR elements are **Requirement**, **Artifact** and **Function**. Design requirements are specifications of some conditions that the product needs to meet, which include functional requirement and non-functional requirement. Functional-requirement can drive the design and lead to some issues, hence what it relates to is **Issue**; meanwhile, non-functional requirement plays the role of design constraint, and what it relates to is **Argument**. As for **Artifact** and **Function**, they make designer understand the design knowledge better as supplements. In this paper, we introduce the DR representation briefly and propose some concepts and relationships for designing DR ontology. For the details of our work about the extended IBIS-based DR representation, please refer to (Li et al. 2013).

#### 4.2. DR ontology

Based on the proposed DR representation, a DR ontology is designed to support the indexing of DR retrieval. The main class hierarchy of the designed DR ontology is shown in Figure 3. First, we create the main concepts according to the extended IBIS-based DR representation which are subclasses of the class **DRElement**. Then we refine the existing classes into subclasses. For **Issue**, **Solution** and **Argument**, we refine them into several types according to their states; for **Function**, the function taxonomy of Hirtz et al. (2002) is introduced as its subclasses; and for **Requirement**, the requirements list

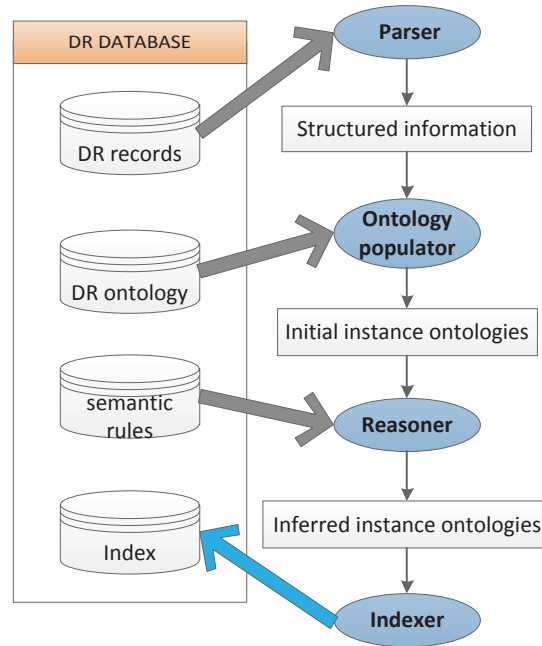


Figure 4. The process of DR ontology-aided indexing.

of Pahl *et al.* (2007) are referenced to enrich its subclasses. Moreover, we add some other concepts according to the basic information of DR records such as author, creation date, etc. Finally, we add relationships between these concepts, e.g. the relationship *support* is added as an object property for concepts *Argument* and *Solution*. As a result, an ontology containing 184 concepts, 26 object properties and 6 datatype properties in DR domain is created.

In order to improve the quality of DR ontology, some classes like *ResolvedIssue* and *InsolubleIssue* are specified to be disjoint, so that an individual (or object) cannot be an instance of more than one of these classes. And some properties like *hasProArg* and *support* are specified to be inverse properties of each other.

## 5. Ontology-aided indexing

In order to improve the effect of DR retrieval, an ontology-aided index is constructed and utilised in this work with the help of the DR ontology described above. The key idea of our ontology-aided indexing is that not only the normal information of the DR record but also the inferred information of the DR record which is generated through ontology reasoning is adopted to index the DR record, so that more semantics can be used to improve the recall and precision of the DR retrieval.

The process of ontology-aided indexing is shown in Figure 4. First, DR records are parsed into structured information. Then, we transform the structured information into ontology individuals by ontology population. After that, we use semantic rules to achieve ontology reasoning and get inferred instance ontologies. Finally, all DR records are indexed with both the normal and inferred information of DR records.

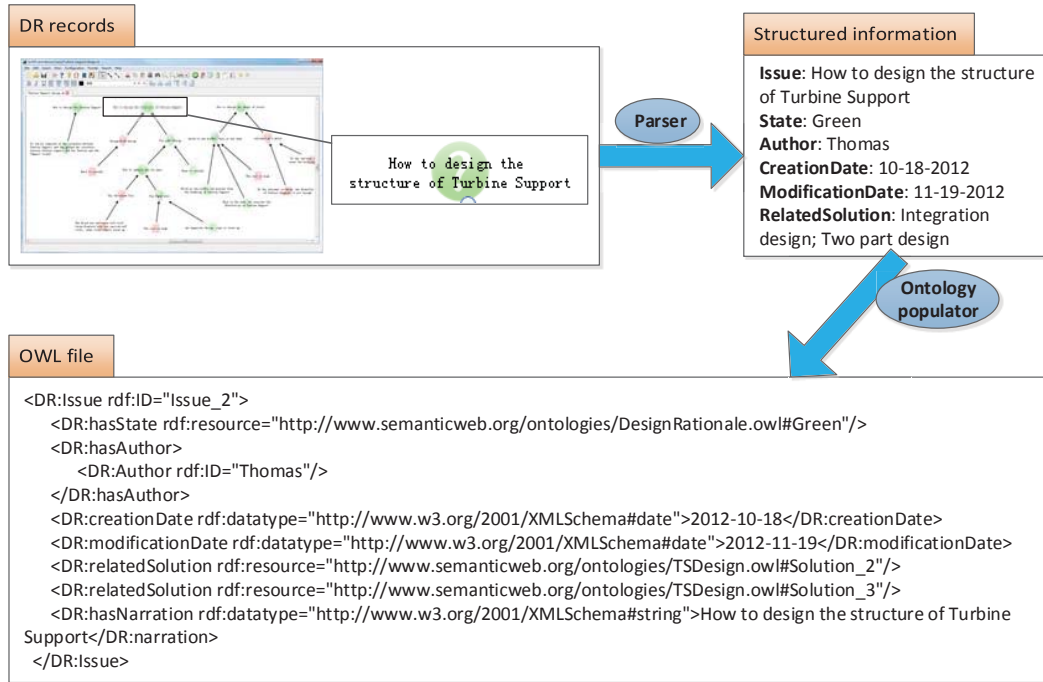


Figure 5. Illustration of ontology population.

### 5.1. *Ontology population*

To achieve the ontology-aided indexing, our first step is to create the DR ontology individuals for all the DR records through ontology population. Ontology population is a knowledge acquisition activity which transforms or maps unstructured, semi-structured and structured data into instance data.

In this work, the ontology population process includes four steps as follows:

- (1) *Create ontology individuals for DR nodes.* An ontology individual is created for each DR node and which class it belongs to depends on the DR node's type.
- (2) *Create datatype properties of DR nodes.* Information inside the DR node such as text and state is added as the ontology individual's properties.
- (3) *Create object properties between DR nodes.* When all the DR nodes are processed by the above two steps, relationships between the DR nodes are then added to the instance ontology as properties of individuals.
4. Create ontology individuals and properties for the DR file. In addition, ontology population is not restricted with the DR nodes. As mentioned above, DR files contain some basic information including authors, creation date, modification date, etc., which are also added to the instance ontology by creating an OWL<sup>3</sup> individual for each of them if they do not already exist in the ontology.

Figure 5 shows the ontology population process starting from DR records ending with OWL individuals.

<sup>3</sup><http://www.w3.org/TR/owl-features/>.

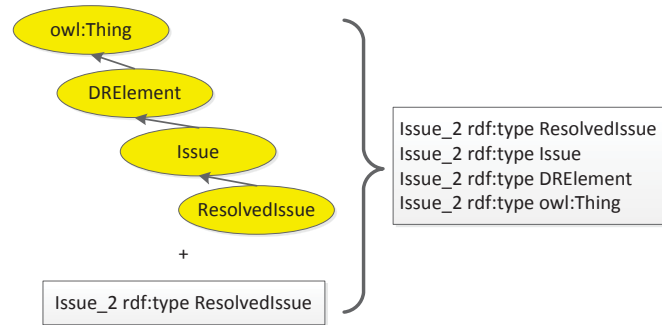


Figure 6. Inferring of an ontology individual.

## 5.2. *Ontology reasoning*

After ontology population is finished, ontology reasoning is conducted on the ontology individuals created through ontology population to achieve the following two purposes: one is to validate the ontology individuals to guarantee their validity; and the other is to obtain the inferred semantic information by ontology reasoning as much as possible that can be used to effectively improve the performance of the DR retrieval. Specifically, two types of ontology reasoning methods are used in this work, namely standard inference services and rule-based reasoning.

The formal specification of Web Ontology Language, OWL, is highly influenced by Description Logics (DLs). OWL-DL is designed to be computationally complete and decidable version of OWL, thus it benefits from a wide range of sound, complete and terminating DL reasoners. For our reasoning module, we use Pellet<sup>4</sup>, an open-source DL-reasoner, which supports all the standard inference services that are traditionally provided by DL reasoners such as consistency checking, concept satisfiability, classification and realization. In addition, Pellet has an implementation of a direct tableau algorithm for a DL-safe rules encoded in SWRL<sup>5</sup> and includes support for some SWRL built-ins.

Standard reference services are used to validate the ontology individuals and obtain certain kinds of inferred information including consistency checking, classification and realization. Consistency checking ensures that there is no contradictory assertion in the ontology, and this is the initial validation method of instance ontologies. We fix the inconsistency before any other reasoning service, since any consequence can be inferred from inconsistency. Fortunately, as our DR ontology is well defined and DR records are captured in a reasonably structured way, there has emerged no inconsistency so far. Using classification and realization, we identify more specific type for each ontology individual. A simple example is given in Figure 6, which shows that the class hierarchy of *ResolvedIssue* is inferred.

Rule-based reasoning is adopted to obtain all the other kinds of inferred information. To illustrate the power of SWRL rules, some examples are given in Table 1. The first three rules are able to infer more specific types for ontology individuals according to DR nodes' states. Although rule (4) and (5) are used to infer more specific types too, they are more complex with more related semantics. Rule (4) means that if an accepted solution has supporting and objecting arguments at the same time, and there is only one

<sup>4</sup><http://clarkparsia.com/pellet/>.

<sup>5</sup><http://www.w3.org/Submission/SWRL/>.

Table 1. Partial SWRL rules defined in DR ontology

---

(1) $\text{Issue}(?i) \wedge \text{hasState}(?i, \text{Yellow}) \rightarrow \text{OpenIssue}(?i)$
(2) $\text{Solution}(?s) \wedge \text{hasState}(?s, \text{Green}) \rightarrow \text{AcceptedSolution}(?s)$
(3) $\text{Argument}(?a) \wedge \text{hasState}(?a, \text{Red}) \rightarrow \text{ObjectToArgument}(?a)$
(4) $\text{AcceptedSolution}(?s) \wedge \text{hasConArgNo}(?s, ?no1) \wedge \text{graterThan}(?no1, 0) \wedge \text{hasProArg}(?s, ?a) \wedge \text{hasProArgNo}(?s, ?no2) \wedge \text{isEqualTo}(?no2, 1) \rightarrow \text{DecisiveArgument}(?a)$
(5) $\text{Solution}(?s) \wedge \text{hasConArgNo}(?s, ?no1) \wedge \text{isEqualTo}(?no1, 0) \wedge \text{hasProArgNo}(?s, ?no2) \wedge \text{greaterThan}(?no2, 0) \rightarrow \text{AcceptedSolution}(?s)$
(6) $\text{ProArgument}(?a) \wedge \text{Solution}(?s) \wedge \text{hasArgument}(?s, ?a) \rightarrow \text{support}(?a, ?s)$
(7) $\text{ConArgument}(?a) \wedge \text{Solution}(?s) \wedge \text{hasArgument}(?s, ?a) \rightarrow \text{objectTo}(?a, ?s)$
(9) $\text{Solution}(?s) \wedge \text{Issue}(?i1) \wedge \text{Issue}(?i2) \wedge \text{respondTo}(?s, ?i1) \wedge \text{leadTo}(?s, ?i2) \rightarrow \text{affect}(?i2, ?i1)$
(10) $\text{Issue}(?i1) \wedge \text{Issue}(?i2) \wedge \text{hasSubIssue}(?i1, ?i2) \rightarrow \text{affect}(?i2, ?i1)$
(11) $\text{Issue}(?i) \wedge \text{Solution}(?s1) \wedge \text{Solution}(?s2) \wedge \text{hasSolution}(?i, ?s1) \wedge \text{hasSubSolution}(?s1, ?s2) \rightarrow \text{hasSolution}(?i, ?s2)$
(12) $\text{Solution}(?2) \wedge \text{Argument}(?a1) \wedge \text{Argument}(?a2) \wedge \text{hasArgument}(?s, ?a1) \wedge \text{hasSubArgument}(?a1, ?a2) \rightarrow \text{hasArgument}(?s, ?a2)$

---

supporting argument, then we can infer that the supporting argument is very important. And the meaning of rule (5) is that if a solution has only supporting arguments, then the solution is an accepted solution, so this rule can be used to validate the DR nodes state. For the last seven rules, either of them infers an implicit relationship between ontology individuals.

Ontology reasoning can be time-consuming when there are many assertions (ABox). However, this issue does not affect our retrieval performance because of the following two reasons: 1) we keep each DR file separate from each other and run the reasoning separately, so a large number of assertions will be hard to appear; 2) all the reasoning tasks, without rules or with rules, are done offline.

### 5.3. Establishing of index

Based on the DR record and its related inferred semantic information, we construct the index of the DR record as follows:

- (1) Normal information is obtained from parsing the DR record. As shown in Table 2, normal information contains what DR record directly shows and some basic information of the DR file such as author, creation date, etc.
- (2) Inferred information is obtained from the instance ontologies of DR after ontology reasoning. Table 3 is an example of inferred information. Note that a new value **ResolvedIssue** is added into the **DRElement** field, and the **LeadToSolution** and **AffectingIssue** fields are also filled using the semantic rules.
- (3) An inverted index is constructed for both of the normal information and the

Table 2. Normal information in index.

Field	Value
DRElement	Issue
NodeID	Issue_2
FilePath	D:/liluye/designData/TSDesign.dr
Author	Thomas
CreationDate	10-18-2012
ModificationDate	11-19-2012
State	Green
RelatedSolution	Integration design, Two part design
RelatedArtifact	Support
LeadToSolution	-
AffectingIssue	-
Narration	How to design the structure of Turbine Support

Table 3. Inferred information in index.

Field	Value
DRElement	ResolvedIssue
LeadToSolution	It can be composed of the interface...
AffectingIssue	How to combine the two part

inferred information. We use Lucene<sup>6</sup> to build this index.

## 6. Flexible querying

In order to support flexible querying which meets users' different requirements, we provide three different ways of query including keyword-based query, natural language input query and DR record based query. For each query mode, a corresponding query processing method is given below.

### 6.1. *Keyword-based query*

Keyword-based query is the most common and people are used to using it. In this work, three steps are provided to process this query. Firstly, spell checking is done to make sure that users' input is correct and some stop words are filtered. Secondly, the keywords are expanded with their synonyms in order to improve the retrieval recall. Specifically, query terms are expanded with synonyms and variants through referring to the WordNet<sup>7</sup>. Thirdly, the retrieval results are classified according to the taxonomy of our DR ontology to enhance the retrieval precision. It is worth noting that all the above-mentioned steps are implemented based on Lucene.

<sup>6</sup><http://lucene.apache.org/>.

<sup>7</sup><http://wordnet.princeton.edu/>.

Table 4. Knowledge needs and corresponding retrieval results.

Interr.	Knowledge needs	Types of retrieval results
<i>How</i>	How a particular part of the product functioned.	Solutions
<i>Why</i>	Why a design is carried out in a particular way.	Arguments
<i>What</i>	What issues to consider.	Issues
<i>When</i>	When issues should be considered.	Requirements & Solutions

## 6.2. Natural language input query

In view that natural language can express people's thinking well and contains more semantic information than keywords, natural language input query is also supported in our DR retrieval system. The expressivity of natural language enables the DR retrieval to take advantage of some relationships in DR ontology. According to the designers' requirements on design knowledge identified by [Ahmed and Wallace \(2004\)](#) and considering the existing NLP technologies, we adopt question-answering strategy to deal with natural language input query, and four types of questions are considered as listed in Table 4.

To effectively handle the four types of questions and get the required answers, five steps are given as follows:

- (1) Use the Stanford parser to parse the natural language question and get the corresponding part-of-speech tagged text, among which the word tagged with [WRB] is used to identify the question type, and verbs and nouns are extracted as keywords  $K$ .
- (2) For *How* questions, the retrieval system will search similar issues with  $K$  and show the corresponding solutions as retrieval results.
- (3) For *Why* questions, similar solutions are searched with  $K$  and the corresponding arguments are shown as results.
- (4) For *What* questions, we search with the  $K$ , and find the issues which are in or related to the initial results.
- (5) For *When* questions, similar issues are searched with  $K$  and the functional requirements or solutions which lead to the issues are shown as retrieval results.

## 6.3. DR record based query

Compared with keywords and natural language, DR record is more structured and the DR semantics involved in it are more abundant, which can help designers better express their knowledge requirements. Considering this, DR record based query is supported in our DR retrieval system as an accurate query mode. In addition, this query mode is imperative for the integrated DR capture and retrieval system. For example, when a designer creates a new DR file during his design work, he inserts an Issue and wants to determine whether there exist solutions responding to this issue in DR database. In this moment, the DR record based query is very helpful for the user.

In this work, two types of DR record based query are provided as shown in Figure 7(a) and Figure 7(b). One is to search similar DR records with a given DR record, and the other is to search the required DR nodes with a given DR record which contains several blank DR nodes. The blank DR nodes represent what users require, and the other DR nodes and relationships of the DR record represent some semantic constrains.

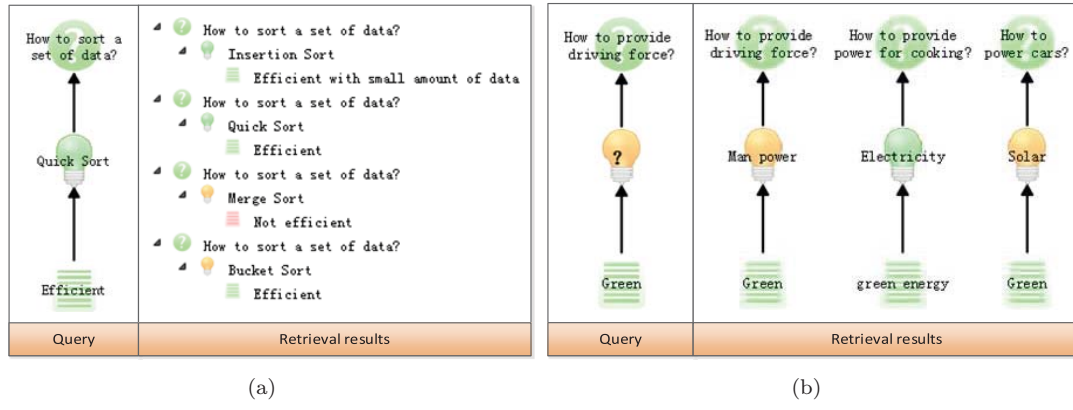


Figure 7. DR record based queries and wanted results: (a) query for similar results, and (b) query for wanted results.

The former is like the traditional retrieval to search similar objects but has the ability to search all the DR nodes and the relationships between them at one time. The latter is like a strengthen version of natural language input query. A natural language question which can be processed by a computer in our work contains only one relationship in DR ontology. However, the relationships contained in the DR record based query shown in Fig 7(b) can be as many as the user wants.

---

**Algorithm 1** An retrieval algorithm for DR record based query

---

```

1: procedure DR_RECORD_BASED_SEARCH(nodeList, pos)  $\triangleright$  nodeList: List of
   nodes ordered by depth-first traversal.  $\triangleright$  pos: No. of current node in nodeList.
2:   if pos = nodeList.size then
3:     output solList  $\triangleright$  solList: List for a feasible solution.
4:     return
5:   end if
6:   node  $\leftarrow$  nodeList.at(pos)
7:   while stack.size > node.depth do
8:     stack.pop
9:   end while
10:  father_sol  $\leftarrow$  stack.top
11:  node_sol_set  $\leftarrow$  getSolution(node, father_sol)  $\triangleright$  get all the possible solutions of
   the current node limited by the father solution.
12:  for all node_sol  $\in$  node_sol_set do
13:    stack.push(node_sol)
14:    solList.add(node_sol)
15:    DR_RECORD_BASED_SEARCH(nodeList, pos + 1)
16:    while stack.size > node.depth do
17:      stack.pop
18:    end while
19:    solList.remove(node_sol)
20:  end for
21: end procedure

```

---

Although DR record based query itself is a little complex, it is not difficult to process. There is no semantic gap between DR record based query and the DR records in DR

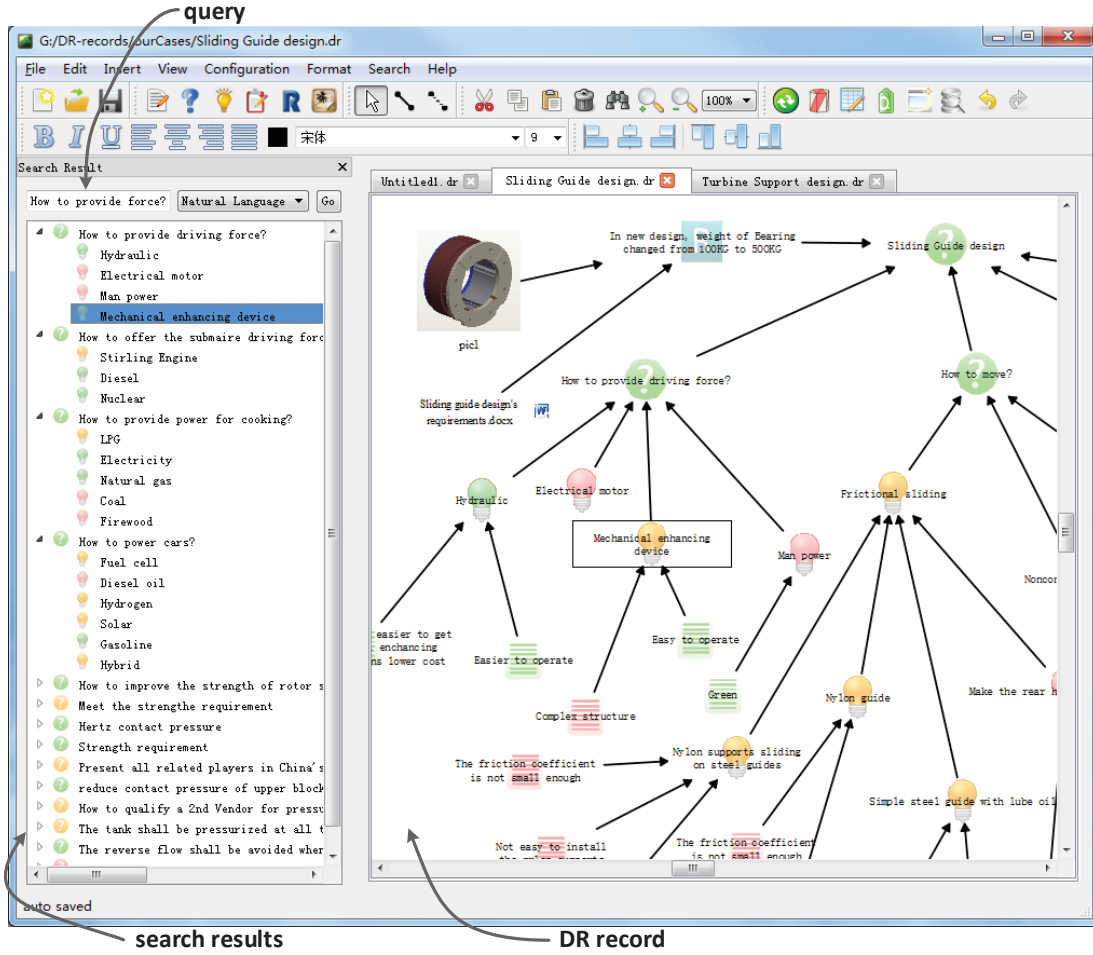


Figure 8. User interface of DR capture and retrieval system.

database, since they use the same representation. As a result, the processing of DR record based query is as the same as ontology-aided indexing described in Fig. 4 except for the final step. First, all the DR records in DR database are processed to a large forest with many trees, and the DR record based query is processed to several subtrees. Then, a tree matching algorithm is adopted to find out similar subtrees as the retrieval results. In addition, our algorithm considers the properties of the edges, and the question mark in Figure 7(b) is processed as a wildcard in our retrieval approach. The pseudocode of our algorithm for DR record based query processing is given in Algorithm 1.

## 7. Implementation and evaluation

### 7.1. System implementation

The proposed DR retrieval approach has been implemented in a multi-module prototype system. The core module for realizing the retrieving function is developed by using Java, while the user interface module (Figure 8) is developed using Qt 4.7.3, which is integrated with our DR capture tool (Li et al. 2013). And the DR records utilised in this research are captured by the DR capture tool which is developed based on the extended IBIS-based

Table 5. Evaluation queries and results for three indexes (precision and recall).

Test queries	INDEX_T		INDEX_S_BR		INDEX_S_AR	
	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.
Q1: provide force ( <i>Resolved Issue</i> )	0.04	1	0.25	1	0.40	1
Q2: sort ( <i>Open Solution</i> )	0.09	1	0.12	1	0.25	1
Q3: install ( <i>Negative Argument</i> )	0.29	1	0.56	1	1	1
Q4: How to provide force?	-	-	0.46	0.99	0.30	1
Q5: How to sort?	-	-	0.63	0.71	0.70	1
Q6: Why not choose merge sort?	-	-	0.07	0.33	0.18	1
Q7: Why do we use gasoline?	-	-	0.15	0.67	0.21	1

DR representation. The three elements from IBIS are given a traffic light status, which refers to DRed (Bracewell *et al.* 2009).

Currently, our DR database for retrieval contains 106 DR records and 1530 DR nodes, most of which are captured by experienced engineering designers. And they are about the design of gas turbine, and the majority of them are just staying in designers' minds instead of being recorded in formal documents. In addition to DR files' basic information, normal information of all the DR records contains 1530 DR nodes' types and 1948 properties. After reasoning with 18 SWRL rules, additional 4419 types and 965 properties are inferred.

As is shown in Figure 8, a DR retrieval with natural language input query is taken as an example, the top left panel left panel shows the natural language input query, the bottom left shows the search results, and the right panel is where the DR record appears. When one line of the results is double-clicked, the corresponding DR record will be shown in the right panel, and the corresponding DR node will be focused.

## 7.2. Evaluation on indexing method

In order to evaluate the retrieval performance of our system, we build three indexes for the DR search, namely INDEX\_T, INDEX\_S\_BR and INDEX\_S\_AR, where the first one is built from the text in each DR node and the latter two are built based on the semantic representation of DR records. Specifically, the second one which is built before reasoning contains only the normal information in DR records and the third one contains the inferred information in addition to the normal information.

The evaluation queries and retrieval results can be seen in Table 5. First of all, consider the first three queries which are keyword-based queries. The precision values are increased obviously as more semantic information captured in the index. Take Q1 as an example, when using INDEX\_T, all types of DR nodes which contain provide force are returned as results. When using INDEX\_S\_BR which contains some basic types of DR nodes like *Issue*, the range of results can be limited to the *Issue* nodes. Furthermore, when using INDEX\_S\_AR which contains some more specific types of DR nodes like *ResolvedIssue*, the range of results are further limited. As a result, the precision values are increased step by step. Its worth noting that we get the inferred types by executing the SWRL rules like the first 5 rules in Table 1. In addition, for the last three queries which are natural language input queries, they are mainly about the relationships between DR nodes. It can be seen that INDEX\_T does not support natural language input query

since it contains no relationship information. Meanwhile, in Table 5 we can also see that the recall values are increased from INDEX\_S\_BR to INDEX\_S\_AR. The reason is that implicit relationships are found out by reasoning with the semantic rules in Tab 1. Specifically, rule (11) affects the results of Q4 and Q5, and rule (12) for Q6. In summary, inferred DR types improve the retrieval precision of keyword-based query and inferred relationships enhance the retrieval recall of natural language input query and DR record based query.

Our evaluation results show that INDEX\_S\_AR has the best retrieval performance, INDEX\_S\_BR has the next best and INDEX\_T has the least. This is because INDEX\_S\_BR contains more semantic information than INDEX\_T does, and much more inferred semantic information is used in INDEX\_S\_AR. As mentioned above, so much inferred information is explored by using only 18 SWRL rules upon the small DR database. It can be expected that the ontology-aided indexing will play a bigger role with better defined ontology, more abundant semantic rules and a larger DR database.

### 7.3. Evaluation on flexible querying

To evaluate the retrieval performance using different query modes, six test cases which correspond to six different users knowledge needs are given in Table 6, so does the corresponding queries under the three query modes. It can be seen that DR record based query can meet all the six knowledge requirements, while natural language query can meet partial of them. In addition, six keyword-based queries are also generated, even though none of them can fully express what users want.

Using the three query modes (mode 1 is keyword-based query, mode 2 is natural language input query, and mode 3 is DR record based query), and under the retrieval setting shown in Table 6, the corresponding retrieval results for the six test cases are given in Table 7. For the first two cases, the retrieval results of mode 2 and mode 3 are the same. That is because either of them contains only one relationship which can be expressed by both mode 2 and mode 3. For case 3 and case 4, the precisions of mode 3 increase obviously comparing with the precisions of mode 2, since both of the cases contain two relationships which can be expressed in mode 3, while only one of the relationships can be expressed in mode 2. For the last two cases, mode 2 cannot meet users knowledge needs, since the relationship in case 5 does not belong to the relationships used in mode 2, and case 6 is like a case based retrieval which is inconsistent with what mode 2 can express. In addition, most of the precisions and recalls of the retrieval using query mode 1 are very small, since mode 1 can barely meet the knowledge needs in the six test cases. It is worth noting that three values of the precisions and recalls in mode 1 are unusually large. After our analysis of the retrieval results, it is a coincidence that the recall of case 1 and the precision of case 2 are so large. However, the recall of case 6 is definitely reasonable since keyword-based query is also like case based query in a way.

The results show that DR record based query is the most powerful query mode which can meet the most knowledge needs and make full advantage of DR information. However, it is also the most complex query mode, though the integration of DR capturing and retrieving makes it much convenient. In addition, keyword-based query is the most common used and easiest way to retrieve. Meanwhile, the retrieval results of the natural language input query directly answer the user instead of searching similar results. With different precisions and different popularities, three different kinds of query modes support flexible querying of our DR retrieval system.

Table 6. Retrieval results for the six test cases under three query modes.

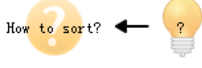
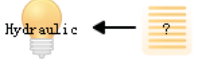



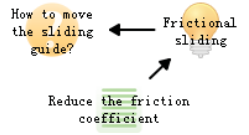
Test cases	Knowledge needs	Keyword-based query (mode 1)	Natural language input query (mode 2)	DR record based query (mode 3)
Case 1	The user would like to see the solutions of sorting numbers	sort ( <i>Solution</i> )	How to sort?	
Case 2	The user would like to see the pros and cons of using hydraulic.	hydraulic ( <i>Argument</i> )	Why use hydraulic?	
Case 3	The user would like to see the pros and cons of using electricity to power a car.	power, car, electricity ( <i>Argument</i> )	Why do we use electricity?	
Case 4	The user would like to see the solutions of providing force considering "green".	provide force, green ( <i>Solution</i> )	How to provide force?	
Case 5	The user would like to see all the solutions considering cost.	cost ( <i>Solution</i> )	-	
Case 6	The user has a solution of moving the sliding guide and can reduce the friction coefficient, and would like to see how it worked in previous projects.	move sliding guide, friction, reduce, friction coefficient	-	

Table 7. Retrieval results for the six test cases under three query modes.

Test cases	Query mode 1		Query mode 2		Query mode 3	
	Pre.	Rec.	Pre.	Rec.	Pre.	Rec.
Case 1	0.29	1	0.46	1	0.46	1
Case 2	1	0.25	1	1	1	1
Case 3	0	0	0.50	1	1	1
Case 4	0.03	0.33	0.05	1	1	1
Case 5	0.07	0.03	-	-	1	1
Case 6	0.02	1	-	-	1	1

## 8. Conclusion and future work

In this paper, a novel DR retrieval approach is presented. It makes following contributions: 1) an ontology-aided indexing method is proposed to index the DR records, through

which the performance of DR search and retrieval can be largely improved; 2) three kinds of query modes are designed to meet different user requirements in terms of knowledge search and retrieval, as opposed to the most commonly used keyword-based query, natural language input query and DR record based query have much more expressivity and take more advantage from the ontology-aided indexing; 3) a DR ontology is created based on the extended IBIS-based DR representation, which contains more domain knowledge and thus can effectively support DR retrieval and reuse. The experimental results show that the proposed DR retrieval approach is better than the traditional keyword-based retrieval, especially on the retrieval precision, and the three kinds of query modes offer different retrieval precisions and suit for different usage scenarios or different users.

In the future, more NLP technologies such as question answering will be applied to fine tune natural language input query. Meanwhile, more extensive experiments are outlined to test the scalability of the proposed approach.

## Acknowledgement

The authors are very grateful to the financial support from the National Science Foundation of China (NO. 61173125).

## References

- Ahmed, S. and Wallace, K.M., 2004. Understanding the knowledge needs of novice designers in the aerospace industry. *Design Studies*, 25 (2), 155–173.
- Bracewell, R.H., *et al.*, 2009. Capturing design rationale. *Computer-Aided Design*, 41 (3), 173–186.
- Burge, J.E. and Brown, D.C., 2008. Software Engineering Using RATIONALE. *Journal of Systems and Software*, 81 (3), 395–413.
- Conklin, J. and Begeman, M.L., 1988. gIBIS: a hypertext tool for exploratory policy discussion. *ACM Transactions on Information Systems (TOIS)*, 6 (4), 303–331.
- De Medeiros, A.P. and Schwabe, D., 2008. Kuaba approach: integrating formal semantics and design rationale representation to support design reuse. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 22 (4), 399–419.
- Hirtz, J., *et al.*, 2002. A functional basis for engineering design: reconciling and evolving previous efforts. *Research in Engineering Design*, 13 (2), 65–82.
- Kara, S., *et al.*, 2012. An ontology-based retrieval system using semantic indexing. *Information Systems*, 37 (4), 294–305.
- Kim, S., Bracewell, R.H., and Wallace, K.M., 2007. Improving design reuse using context. In: *Proceedings of the 16th International Conference on Engineering Design (ICED'07)*, Paris, France.
- Kim, S., Bracewell, R.H., and Wallace, K.M., 2005. A framework for design rationale retrieval. In: *Proceedings of the International Conference on Engineering Design (ICED'05)*, Melbourne, Australia.
- Kunz, W. and Rittel, H.W.J., 1970. *Issues as elements of information systems*. Vol. 131. Institute of Urban and Regional Development, University of California at Berkeley, Berkeley, CA.
- Li, L., Qin, F., and Gao, S., 2013. An extended design rationale representation for sup-

- porting retrieval and reuse of design knowledge. *Journal of Computer-Aided Design & Computer Graphics*, 25 (10), 1514–1522.
- Liang, Y., *et al.*, 2012. Learning the “why”: discovering design rationale using text mining - an algorithm perspective. *Computer-Aided Design*, 44 (10), 916–930.
- Liang, Y., *et al.*, 2010. Interactive interface design for design rationale search and retrieval. *In: Proceedings of ASME/IDETC&CIE Conference*, Montreal, Quebec, Canada.
- Lim, S.C.J., Liu, Y., and Lee, W.B., 2010. Multi-facet product information search and retrieval using semantically annotated product family ontology. *Information Processing & Management*, 46 (4), 479–493.
- Lim, S.C.J., Liu, Y., and Lee, W.B., 2011. A methodology for building a semantically annotated multi-faceted ontology for product family modelling. *Advanced Engineering Informatics*, 25 (2), 147–161.
- Liu, Y., *et al.*, 2010. A new design rationale representation model for rationale mining. *Journal of Computing and Information Science in Engineering*, 10 (3), 031009.
- López, C., Cysneiros, L.M., and Astudillo, H., 2008. NDR ontology: sharing and reusing NFR and design rationale knowledge. *In: Proceedings of the First International Workshop on Managing Requirements Knowledge (MARK'08)*, 1–10.
- MacLean, A., *et al.*, 1991. Questions, options, and criteria: elements of design space analysis. *Human-Computer Interaction*, 6 (3-4), 201–250.
- McCall, R.J., 1991. PHI: a conceptual foundation for design hypermedia. *Design Studies*, 12 (1), 30–41.
- Pahl, G., Wallace, K., and Blessing, L., 2007. *Engineering design: a systematic approach*. Vol. 157. Springer.
- Qin, F., Li, L., and Gao, S., 2012. A survey of design rationale. *Journal of Computer-Aided Design & Computer Graphics*, 24 (10), 1283–1293.
- Regli, W.C., *et al.*, 2000. A survey of design rationale systems: approaches, representation, capture and retrieval. *Engineering with Computers*, 16 (3-4), 209–235.
- Rockwell, J., *et al.*, 2009. A decision support ontology for collaborative decision making in engineering design. *In: Proceedings of the 2009 International Symposium on Collaborative Technologies and Systems*, 1–9.
- Wang, H., Johnson, A.L., and Bracewell, R.H., 2009. Supporting design rationale retrieval for design knowledge re-use. *In: Proceedings of the 17th International Conference on Engineering Design (ICED'09)*, Stanford, California, USA, 335–346.
- Wang, H., Johnson, A.L., and Bracewell, R.H., 2012. The retrieval of structured design rationale for the re-use of design knowledge with an integrated representation. *Advanced Engineering Informatics*, 26 (2), 251–266.
- Zhang, Y., *et al.*, 2013. A semantic representation model for design rationale of products. *Advanced Engineering Informatics*, 27 (1), 13–26.