Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 53 (2016) 50 – 55

www.elsevier.com/locate/procedia

The 10th International Conference on Axiomatic Design, ICAD 2016

Innovative design thinking for breakthrough product development

Stephen Lu[a*] and Ang Liu[b]

[a]*Department of Industrial and System Engineering, University of Southern California, Los Angeles, 91803, USA*
[b]University of New South Wales, Sydney 2052, Australia

* Corresponding author. *E-mail address:* sclu@usc.edu

Abstract

Concept generation is the most critical task in breakthrough product development. This paper presents an Innovative Design Thinking (IDT) framework that models concept generation as a proposition-making activity according to the formation definition of logic propositions. IDT formalizes designers' verbal statements as either analytic or synthetic propositions through a cyclic operation of "specify-ideate-validate" at each abstraction level to generate a design concept which is logically feasible, functionally simple, and physically certain. Then, IDT guides designers through a zigzagging process which repeats the same cyclic operation at progressively less abstract levels to complete concept generation. Details of this cyclic operation and the zigzagging process are explained in this paper with an illustrative example presented.

Keywords: Conceptual Design; Concept Gneration; Breakthrough Innovation

1. Introduction

Breakthrough products are developed by discovering unmet customer needs (CN), choosing exciting functional requirements (FR), ideating innovative concepts in terms of design parameters (DP), and finally optimizing design performance via process variables (PV) under constraints. Concept generation, where relationships between FR to DP are established, is the most important and challenging phase in breakthrough product development. It is important because the designer's creativity at this early phase will ultimately determine the product's quality at the later phase. It is challenging because, unlike analysis which evaluates existing options in a closed form, synthesis must create new concepts that never existed before. The reasoning activity of synthesizing multiple entities towards something new is very different from analyzing the performance of things in existing. The latter is well-supported by many modeling, simulation, and optimizing tools; whereas the former is poorly understood and ill-practiced largely based on the designer's experience [1]. This is the main hindrance of creative concept generation in design practice, which limits the success of breakthrough product development.

When designers brainstorm ideas at the beginning of product development, they typically express initial opinions and make preliminary suggestions using some verbal statements. Spoken language is the most common, and sometimes the only, mediator used by design teams during concept generation. Unfortunately, human language is inherently vague and often loosely expressed by people with different meanings and interpretations, making it difficult to use to generate, represent, and organize good design concepts. It is clear that, if these informal statements could be structured formally, such that their embedded meanings can be made explicit and evaluated objectively, then they will be more useful for developing breakthrough products. This is the motivation behind our Innovative Design Thinking (IDT) research to develop a framework that guides the designer to formulate their informal verbal statements as formal logic propositions to perform analysis and synthesis activities in new product development.

IDT organizes breakthrough product development into three consecutive stages: Functional Design (i.e., from CN to

FR), Conceptual Design (i.e., from FR to DP), and Technical Design (i.e., from DP to PV). The Conceptual Design Stage is further divided as two iterative phases: the Concept Generation Phase and the Concept Improvement Phase. IDT treats concept generation as an organized "proposition-making" activity according to the formal definitions of proposition in logic [2-4]. As the designer proposes various DPs to satisfy the chosen FRs during the Concept Generation Phase, IDT guides the designer to organize his verbal statements as logic propositions (i.e., structured statements of subject-predicate pairs), so that various ideas proposed by different designers can be combined, compared, and selected systematically towards better design outcome. Based on logic definitions, two types of propositions are adopted in IDT: analytic and synthetic propositions. The logic orthogonality (or mutual exclusiveness) between analytic and synthetic propositions results in a two dimensional reasoning roadmap to guide IDT's Concept Generation Phase, which is carried out via three consecutive Steps. First, in the Formation Step, IDT guides the designer to make various analytic and synthetic propositions through a closed loop of "specify-ideate-validate" to form an initial option space which only consists of logically feasible concepts. Next, in the Organization Step, IDT adopts the Independence Axiom from the Axiomatic Design Theory (ADT) to classify those logically feasible concepts into uncoupled, decoupled, and coupled categories according to their degree of functional dependency [5-6]. Finally, in the Selection Step, IDT guides the designer to use various criteria to choose the best concept which is not only logically feasible and functionally simple but also physically certain. After completing these steps at a certain level, IDT then guides the designer to repeat the same steps at progressively more detailed levels until a tangible design concept is obtained or available design resources are exhausted.

From a practical viewpoint, IDT can be seen as a "hybrid" approach between the "decompose-generate-compose" cycle prescribed by the Analytical Target Cascading (ATC) [7-8], and the "layer-by-layer" zigzagging process suggested by the ADT [5-6]. As a result, IDT is most suitable for design practices in between the two extreme cases of analysis-based routine designs (by ATC) and synthesis-focused creative designs (by ADT). IDT can guide the designer to systematically alternate between analysis reasoning and synthesis reasoning to support a wide range of design tasks to achieve a good balance between creativity and practicality.

The focus of this paper is on IDT's Concept Generation Phase during the Conceptual Design Stage, specifically, the Concept Formation Step. Due to the space limitation, details of the Concept Organization and Selection Steps, which are similar to the process of applying the two design axioms prescribed in ADT, will not be elaborated in this paper. Interested readers are encouraged to study relevant ADT publications for a thorough understanding [9-10].

2. Theoretical underpinnings of IDT

Innovative Design Thinking (IDT) is not a single decision method based on certain fixed algorithms to optimize the design result; nor it is an exact design theory that imposes a normative view toward the design process. Rather, it is a domain-independent framework based on well-established definitions in logic, epistemology, and philosophical studies. It draws a set of relevant decision methods and design theories under a single framework to support early stage design. The six theoretical building-blocks of IDT are briefly summarized below.

(A) Reasoning: IDT models concept generation as a "proposition-making" activity based on the formal definition of proposition in logic. As designers propose different ideas of how to satisfy the targeted FR, IDT guides them to formulate their proposals as analytical and synthetic propositions, so that an initial space of logically feasible options can be formed for further comparison and evaluation.

(B) Representation: IDT represents a design concept as logic associations between a set of FR and DP entities, resulted from making analytic propositions within a hierarchy and making synthetic propositions across two separate hierarchies. Such a two dimensional representation scheme is similar to that of the Axiomatic Design Theory by Suh [5-6]. However, IDT uses the formal definition of logic propositions as its theoretical foundation to guide the ideation of design concepts, so that they can be better organized and compared systematically later [11].

(C) Operation: IDT prescribes a "specify-realize-validate" cycle as the basic operation at each abstraction level in order to form a space of logically feasible concepts. The cyclic IDT operation proposes "specified-by" and "realized-by" logic relationships, and then validates the proposed concepts with "part-of" and "means-of" logic relationships to complete the cycle of concept formation. The analytic-synthetic distinction together with the above closed-loop operation cycle enable IDT to systemically guide synthesis and analysis activities, which are often performed arbitrarily in current design practice [9].

(D) Complexity: Based on the assertion that design concepts with less functional complexity are more ideal [12-14], IDT employs the "functional dependency" property of proposed concepts, which can be revealed by different logic associations between FRs and DPs established from making synthetic propositions, to identify the functionally simple options. This notion of functional dependency in IDT is directly adopted from the Independence Axiom of Suh's Axiomatic Design Theory and Design Complexity Theory [12].

(E) Certainty: Among those logically feasible and functionally simple options, IDT further asserts that the best concept is the one with the highest estimated physical certainty. Accordingly, IDT suggests to use various estimation methods to rank-order and select the most physically certain concept. Many existing methods, such as Quality Function Deployment [15], the Information Axiom of the Axiomatic Design Theory [5], Analytic Hierarchical Process [16], etc., can be used to support concept selection based on probability estimation.

(F) Process: After completing the above cyclic operation of "specify-realize-validate" at a certain abstraction level, IDT follows a zigzagging pathway to repeat the same operation at progressively lowers abstraction layer to further evolve the ideated concepts with more details and specifics. This zigzagging process can be considered as a special kind of "coevolution" that is widely studied in biology [17], software engineering [18], and design studies [19-20]. For example, a similar zigzagging pathway was also prescribed by the Axiomatic Design Theory [5].

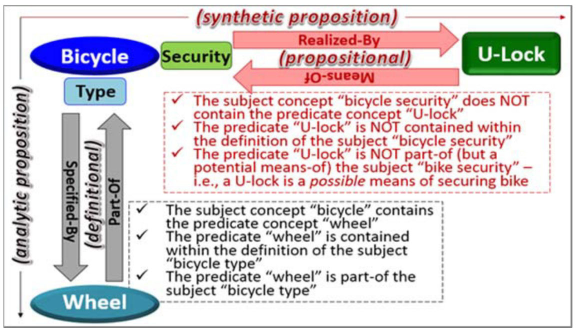
Based on the above "reasoning" and "representation" schemes (see Building Blocks A and B) at each operation level, IDT guides the designer to complete the Concept Generation Phase through the following three consecutive steps:

- The Concept Formation Step: establish a well-defined space of logically feasible options by performing a cyclic operation of "specify-realize-validate" between FR and DP hierarchies (see Building Block C).
- The Concept Organization Step: categorize all the logically feasible options derived in last step into uncoupled, decoupled, and coupled concepts, according to their degree of functional complexity which is determined by the logic associations between FR and DP (see Building Block D).
- The Concept Selection Step: select a particular design concept, from the above logically feasible and functionally simple options, which is estimated to have the highest certainty (or probability of success) in terms of physical implementation (see Building Block E).

After completing the above three steps, the designer will arrive at a particular design concept which is (1) logically feasible, (2) functional simple, and (3) physically certain. IDT then guides the designer to repeat the same procedure at the next operation "level" (i.e., the lower abstraction "layers") with a zigzagging process (see Building Block F) to generate more details of the ideated concept. The zigzagging process continues until a desirable level of details is obtained or available resources (e.g., time, budget, etc.) is exhausted. Note that, during the IDT Concept Formation Step, the cyclic operation of "specify-realize-validate" is performed at a certain operation "level," which consists of two adjacent abstraction "layers" defined by the Axiomatic Design Theory. Reader should pay special attentions to the important difference between IDT's operation "level" and ADT's abstraction "layer" when performing zigzagging (see Section 4 for details).

3. Use IDT to Generate a Preliminary Design Concept at a Specific Operation Level

Among the three steps of the IDT Concept Generation Phase, the Concept Formation Step is the most important and challenging one. Typically, a project begins with designers using spoken language to express their ideas of how to satisfy the targeted FR under given constraints. A "proposition" is a particular type of structured sentence, which declares (i.e., affirm or deny) a specific kind of relational association between the "subject" and the "predicate" of that sentence. The "subject" indicates what the sentence is about, and the "predicate" tells something about the said "subject". For example, the sentence "bicycle has two wheels" is a proposition that affirms the predicate (i.e., "two wheels") of the subject (i.e., "bicycle"); in other words, the predicate "two wheels" specifies something about the subject "bicycle". When a proposition is made, the so stated "something" can establish two different kinds of relationships (e.g., logic association) between the "subject" and the "predicate" in the sentence. In the study of logic, two types of propositions are clearly differentiated according to different kinds of relationships established. Specifically, an analytic proposition is a sentence whose predicate is "contained-within" the subject; whereas a synthetic proposition is a sentence whose predicate is "not-contained-within" the subject [2]. Analytic propositions can be viewed as stating certain "known" facts that are familiar to the designers or well accepted by the community, and therefore should be treated as "definitional". In contrast, synthetic propositions can be viewed as making some tentative suggestions, hence are "propositional" and requires further validation for their acceptance. For example, "bicycle has two wheels" is an analytic proposition, because the predicate of "two wheels" is contained within the subject of "bicycle". This analytic proposition is an affirmative sentence, which states a known fact (or a commonly accepted definition) about the bicycle. On the other hand, "bicycle is unsafe in busy traffic" is a synthetic proposition because the predicate of "unsafe in busy traffic" is not necessarily contained within the subject of "bicycle". But rather, this synthetic proposition is a suggestive sentence, which merely states a condition, an opinion, a proposal, or a judgment, whose truthfulness is to be validated by further evidence before it can be accepted as a known fact.


(synthetic proposition)
Bicycle
Security
Realized-By
(propositional)
Means-Of
U-Lock
Type
(analytic proposition)
Specified-By
(definitional)
Part-Of
Wheel
✓ The subject concept "bicycle security" does NOT contain the predicate concept "U-lock"
✓ The predicate "U-lock" is NOT contained within the definition of the subject "bicycle security"
✓ The predicate "U-lock" is NOT part-of (but a potential means-of) the subject "bike security" – i.e., a U-lock is a *possible* means of securing bike
✓ The subject concept "bicycle" contains the predicate concept "wheel"
✓ The predicate "wheel" is contained within the definition of the subject "bicycle type"
✓ The predicate "wheel" is part-of the subject "bicycle type"
Fig.1. Examples of analytic and synthetic propositions

In IDT, the logic definitions of analytic and synthetic propositions are used to mark and characterize different kinds of design relationships between the subject and the predicate to support concept generation. Making an analytic proposition creates a "part-of" design relationship, which states that the subject entity is "specified-by" the predicate entity. In other words, we can say that the predicate is a "part-of" the subject because the former is contained within the latter. In contrast, since logic clearly defines that the predicate is not contained within its subject, making a synthetic proposition leads to a "means-of" design relationship, which suggests that the subject entity could possibly be "realized-by" its predicate entity. In other words, the predicate is proposed to be a "means-of" achieving the subject; but the former is not contained within the latter. Unlike analytic propositions in which the subject and its predicates are of the same kind and linked by the "part-of" or "specified-by" relationships, the subject and predicates of synthetic propositions are of different kinds and linked by “means-of” or “realized-by” relationships. Therefore, analytic and synthetic propositions must be organized differently in separated hierarchies as will be explained next. For examples (See Figure 1), “bicycle type is specified-by the wheel” is an analytic proposition which states a commonly accepted fact that the wheel (as the predicate) is a “part-of” the definition of a bicycle (as the subject). Whereas, “bicycle security is realized-by U-lock” is a synthetic proposition, which suggests that a U-lock (as the predicate) could be a possible “means-of” securing bicycle (as the subject), while there are many other possible means (e.g., bicycle guard, cable lock, combination lock) to secure bicycles. Since this suggestive statement made as a synthetic proposition is merely a proposed idea, neither a universally accepted fact nor a commonly agreed definition, of achieving bicycle security, it must be validated by other independent evidences (e.g., past records of bicycle thefts) before its acceptance. The above analytic and synthetic propositions concerning a bicycle are illustrated in Figure 1.

When the designer make analytic propositions repeatedly to create many “specified-by” relationships downwards (or “part-of” relationships upwards), an ontological hierarchy of the subject (called the “parent” layer of a hierarchy) is established to include all its corresponding predicates (called the “children” layer of a hierarchy). According to the study of ontology, such a hierarchical structure can be constructed as one of two kinds, taxonomy or meronomy [21]. A taxonomy hierarchy defines the “kind-of” relationship, which is often viewed as a “is-a” link in the Objective Oriented Programming (OOP) modeling [21] or the “extends” operator in the Java programming [22]. On the other hand, a meronomy hierarchy defines the “part-of” relationship, which is commonly viewed as a “has-a” link in the Objective Oriented Programming (OOP) or the “encapsulation” operator in the Java programming [22].

Different from analytic propositions, making a synthetic proposition creates a “realized-by” relationship (or a “means-of” relationship, reversely) between the subject and predicate (e.g., where the proposed ideas, suggestions must be further validated before acceptance). Unlike analytic propositions whose subject and predicate can be organized in a single parent-children hierarchy to accommodate the “specified-by” (or “part-of”) relationships, the “realized-by” (or “means-of”) relationships resulted from synthetic propositions cannot be organized hierarchically in a single hierarchy. This is because the predicates of synthetic propositions are “not” contained within their subjects logically. In other words, since these subject and predicate entities do not belong to the same family, according to the strict definition of a hierarchy, they cannot be organized simply as the parent-children relationships within a single hierarchy. Nonetheless, all the predicate entities of synthetic propositions can be organized among themselves in a separate hierarchy (herein refer to as the 2nd hierarchy), corresponding to the “realized-by” (or “means-of”) relationships linked back to their subjects (i.e., the parent entities) in the 1st hierarchy.

The vertical associations between entities of the same kind within the FR (or DP) hierarchy can help designers to manage

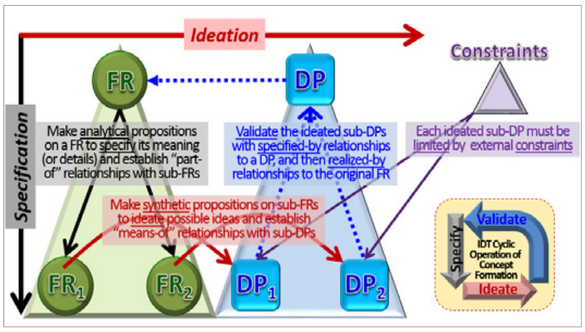

Ideation
Constraints
FR
DP
Specification
Make analytical propositions on a FR to specify its meaning (or details) and establish “part-of” relationships with sub-FRs
Validate the ideated sub-DPs with specified-by relationships to a DP, and then realized-by relationships to the original FR
Each ideated sub-DP must be limited by external constraints
Make synthetic propositions on sub-FRs to ideate possible ideas and establish “means-of” relationships with sub-DPs
FR1
FR2
DP1
DP2
Validate
Specify
IDT Cyclic Operation of Concept Formation
Ideate

Fig.2. Cyclic operation of specify-ideate-validate in IDT

concept generation at different levels of abstraction. The horizontal associations between entities of different kinds across two separate hierarchies reveal the functional dependency information that allows the designer to focus on functionally simple options. Similar to the schematic diagram of electronic circuits, such functional dependency information between FR and DP hierarchies derived in IDT can be seen as the “functional schematics” of a design concept. The creation of logic associations within one hierarchy via analytic propositions and across two hierarchies via synthetic propositions is carried out by the “specify-realize-validate” operation in IDT. This cyclic operation, as indicated in Figure 2, is intended to improve those model-based analysis and heuristics-based synthesis reasoning activities which are often performed in an ad-hoc manner in practice [17].

The above cyclic “specify-ideate-validate” operation forms a “closed-loop” to ensure that all ideated concepts are logically consistent. In order words, a finite space containing only logically feasible options is formed by a chain of logically verifiable statements: a target FR that is specified-by FR_1 and a FR_2 can be realized-by DP_1 and a DP_2, which are part of a main DP that becomes a possible mean-of the original FR. Not that, there are still multiple, but finite, number of logically feasible options contained in this space; and they must be categorized further according to their functional complexity in the IDT Concept Formation Step next. Finally, a particular concept, which is both functionally simple and physically certain, will be selected during the final IDT Concept Selection Step to complete the concept generation task at this operation level.

4. Use IDT to Develop More Details of the Generated Concept via a Zigzagging Process

The completion of the above Concept Formation, Organization, and Selection Steps generates a specific design concept represented by a two dual-hierarchy structure of logic associations (or functional schematics) between FRs and DPs. This constitutes the “ZIG” at a particular operation level in IDT. The same procedure must now be repeated at the next operation level so that more details and specifics of the generated concept can be systemically developed. To do this, the focus of concept generation must now be switched from DP back to FR at the next operation level (or, according to the ADT terminology, the next two abstraction layers with progressively more details). Carrying backwards a concept with its downstream DPs at one operation level to refocus on the upstream FR at the next operation level is called the "ZAG" in IDT. After ZAG back to the functional domain, the same steps of Concept Formation, Organization, and Selection are repeated via the ZIG to add more details to the concept. The following is a sequence of all decision activities occurred in the IDT framework.

1) The Functional Design Stage (CN-FR, to frame a design opportunity)
   - A). Identify an unmet customer need (CN) and clarify a set of design constraints
   - B). Choose a FR as the design target to satisfy the CN, and determine its design range
2) The Conceptual Design Stage (FR-DP, to generate and to improve a design concept to size the design opportunity)
   - A). The *Concept Generation Phase* (to generate a preliminary design concept)
     - i). "ZIG" across two hierarchies at a specific operation level (which is composed of 2 abstraction layers)
       - a) The *Concept Formation* Step (to establish a limited space of *logically feasible* options via the cyclic "specify-ideate-validate" operation)
         - 1) Specification: to declare details of the FR
           - ⇒ Make analytic propositions on the FR to establish specified-by relationships with its sub-FRs
         - 2) Ideation: to suggest means for each sub-FR
           - ⇒ Make synthetic propositions on each sub-FR to establish realized-by relationships with its corresponding sub-DPs
         - 3) Validation: to verify the above sub-DPs
           - ⇒ Make reversed analytic propositions on each sub-DPs to create part-of relationships with a DP
           - ⇒ Make reversed synthetic propositions on the above DP to create means-of relationship with the original FR
       - b) The *Concept Organization Step* (to categorize the logically feasible options according to their functional complexity, which is determined by the synthetic propositions from FR to DP)
         - 1) Ideality Statement: the best design is the functionally simplest one
           - ⇒ Examine the "realized-by" relationships to reveal the degree of functional couplings
         - 2) Categorization Strategy: represent functional complexity by the design matrix [5]
           - ⇒ Form the diagonal, full, and triangular design matrix
       - c) The *Concept Selection Step* (to choose a concept from the above functionally simple ones, which is estimated to have the highest *physical certainty* in terms of implementation)
         - 1) Probability Definition: apprise the likelihood of downstream successes
           - ⇒ Estimate the probability of success for every possible concept
         - 2) Selection Method:
           - ⇒ Rank-order all estimated probabilities to select the most physically certain concept
     - ii). "ZAG" to carry downwards the above generated design concept into the next IDT operation level
     - iii). "ZIG" again across two hierarchies to instantiate the design concept at new abstraction layers
     - iv). "ZAG" again to carry downwards the above design concept into the next IDT operation level
     - v). Repeat the above "ZIG-ZAG" until design resources are exhausted, or the generated design concept is concrete enough for implementation
   - B). The *Concept Improvement Phase* (to make the above generated, or an existing, design concept simpler and/or with higher quality)
3) The Technical Design Stage (DP-PV, to optimize performance of the generated/improved design concept)

5. An illustrative example of IDT

This section presents an illustrative example of how to use IDT to guide designers through various proposition-making activities and zigzagging process step-by-step (see Figure 3) to generate a new product concept which satisfies the FR of "to secure bikes in city". For easy reference, the explanations will follow the same itemized scheme listed in Section 4 above, and the cycled numbers (1-23) in Figure 3 are used to refer to the specific IDT activities as follows:

Step (1) and (2): two analytic propositions were made on the subject FR (to secure bikes in city), which created two specified-by relationships with two resulting predicates FR_1 (to join bike frame with a grounded pole) and FR_2 (to pin two wheels to the bike frame).

Step (3) and (4): two synthetic propositions were made based on the two sub-FRs as the subject, leading to two realized-by relationships with two resulting sub-DPs: DP_1 (a bike frame with a folding lock) and DP_2 (immobilizer pins for two wheels) as the predicates.

Step (5), (6), and (7): two reversed analytic propositions were made on DP_1 and DP_2 as the subjects, resulting in two part-of relationships converging an integrated DP (a "bike-is-the-lock" system) as the predicate. Then, one reversed

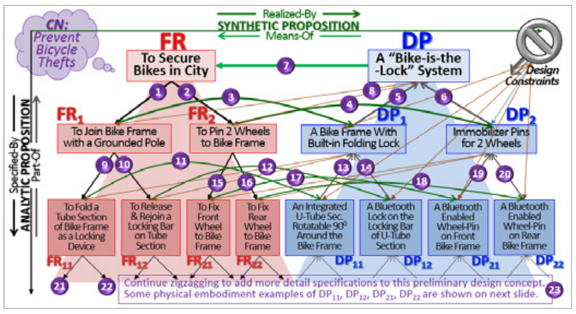

CN: Prevent Bicycle Thefts
Realized-By
SYNTHETIC PROPOSITION
Means-Of
FR
To Secure Bikes in City
DP
A "Bike-is-the -Lock" System
Design Constraints
Specified-By
ANALYTIC PROPOSITION
Part-Of
FR1
To Join Bike Frame with a Grounded Pole
FR2
To Pin 2 Wheels to Bike Frame
DP1
A Bike Frame With Built-in Folding Lock
DP2
Immobilizer Pins for 2 Wheels
To Fold a Tube Section of Bike Frame as a Locking Device
To Release & Rejoin a Locking Bar on Tube Section
To Fix Front Wheel to Bike Frame
To Fix Rear Wheel to Bike Frame
An Integrated U-Tube Sec. Rotatable 90° Around the Bike Frame
A Bluetooth Lock on the Locking Bar of U-Tube Section
A Bluetooth Enabled Wheel-Pin on Front Bike Frame
A Bluetooth Enabled Wheel-Pin on Rear Bike Frame
FR11
FR12
FR21
FR22
DP11
DP12
DP21
DP22
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
Continue zigzagging to add more detail specifications to this preliminary design concept.
Some physical embodiment examples of DP11, DP12, DP21, DP22 are shown on next slide.

Fig.3. Functional schematics of the "bike-is-the-lock" system

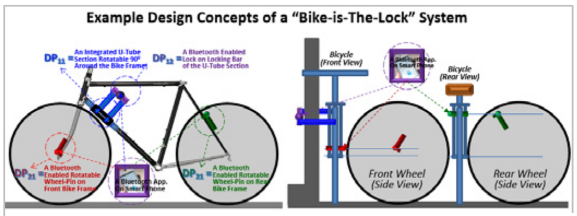

Example Design Concepts of a "Bike-is-The-Lock" System
DP11 = An Integrated U-Tube Section Rotatable 90° Around the Bike Frame
DP12 = A Bluetooth Enabled Lock on Locking Bar of the U-Tube Section
DP21 = A Bluetooth Enabled Rotatable Wheel-Pin on Front Bike Frame
A Bluetooth App. On Smart Phone
DP21 = A Bluetooth Enabled Rotatable Wheel-Pin on Rear Bike Frame
Bicycle (Front View)
A Bluetooth App. On Smart Phone
Bicycle (Rear View)
Front Wheel (Side View)
Rear Wheel (Side View)

Fig.4. Embodiment design of the "bike-is-the-lock" system

synthetic proposition was made on this DP as the subject, leading to a mean-of relationship back to the original FR (to secure bikes in city) as the predicate.

Step (8): while carrying out (5), (6), and (7) above, designers must make sure that design range of the two sub-FRs are within that of the proposed DP. Hence, they often need to iterate through steps (1) to (8) multiple times to validate consistence of the closed-loop cycle.

The steps (1) to (8) constitute a "ZIG" from FR to DP at the first operation level. Next, designers will "ZAG" into the next operation level to carry on the following steps to develop more details of the generated concept.

Step (9), (10), (11), (12), (13), and (14): designers focus on the FR_1 (to join bike with a grounded pole) and repeat the same steps of (1) to (8) on this sub-FR.

Step (15), (16), (17), (18), (19), and (20): designers focus on the FR_2 (to pin two wheels to bike frame) and repeat the same steps (1) to (8) on this sub-FR.

The above steps of (9) to (20) constitute a "ZIG" from FR to DP at the second operation level. As a result, the following four sub-DPs were generated: DP_{11}: An Integrated U-tube section rotatable 90 degree around the bike frame; DP_{12}: A Bluetooth lock on bike's locking bar of the U-tube section; DP_{13}: A Bluetooth-enabled wheel-pin mechanism on the front bike frame; and DP_{14}: A Bluetooth-enabled wheel-pin mechanism on the rear bike frame

The above process completes the Concept Generation Phase of the Conceptual Design Stage in IDT, and it results in a preliminary design concept to be further improved. The generated concept is instantiated by the above four sub-DPs, which together compose a new "bike-is-the-lock" system. An example embodiment of this design concept for the new system is illustrated in Figure 4, where (A) and (B) each shows the unlocked and locked state of the system.

6. Conclusion

Traditionally, designers often express concept ideas as various verbal statements, leading to an ad-hoc process that hinders the innovativeness of product development. This paper presents an Innovative Design Thinking (IDT) framework, which formalizes those informal verbal statements as more structured analytic and synthetic propositions, so that their implied meanings can be systematically organized and compared to yield better design concepts. Both synthesis and analysis reasoning are supported by the IDT framework, leading to design concepts that are logically feasible, functionally simple, and physically certain.

References

[1] Yoshikawa H. Design philosophy: the state of the art. CIRP Annals-Manufacturing Technology. 1989 Dec 31;38(2):579-86.

[2] Kant I, Guyer P. Critique of pure reason. Cambridge University Press; 1998.

[3] Hanna R. Kant and the foundations of analytic philosophy. Oxford: Clarendon Press; 2001 Jan.

[4] De Jong WR. The analytic-synthetic distinction and the classical model of science: Kant, Bolzano and Frege. Synthese. 2010 May 1;174(2):237-61.

[5] Suh NP. Axiomatic Design: Advances and Applications (The Oxford Series on Advanced Manufacturing).

[6] Suh NP. Axiomatic design theory for systems. Research in engineering design. 1998 Dec 1;10(4):189-209.

[7] Kim HM, Rideout DG, Papalambros PY, Stein JL. Analytical target cascading in automotive vehicle design. Journal of Mechanical Design. 2003 Sep 1;125(3):481-9.

[8] Michalek JJ, Feinberg FM, Papalambros PY. Linking marketing and engineering product design decisions via analytical target cascading. Journal of Product Innovation Management. 2005 Jan 1;22(1):42-62.

[9] Liu A, Lu SC. Alternation of analysis and synthesis for concept generation. CIRP Annals-Manufacturing Technology. 2014 Dec 31;63(1):177-80.

[10] Lu SC, Liu A. Abductive reasoning for design synthesis. CIRP Annals-Manufacturing Technology. 2012 Dec 31;61(1):143-6.

[11] Lu S, Liu A. A logic-based foundation of axiomatic design. InProceedings of ICAD, 6th International Conference on Axiomatic Design, Daejeon, South Korea, March 2011 (pp. 30-31).

[12] Suh NP. Complexity in engineering. CIRP Annals-Manufacturing Technology. 2005 Dec 31;54(2):46-63.

[13] ElMaraghy W, ElMaraghy H, Tomiyama T, Monostori L. Complexity in engineering design and manufacturing. CIRP Annals-Manufacturing Technology. 2012 Dec 31;61(2):793-814.

[14] Lu SC, Suh NP. Complexity in design of technical systems. CIRP Annals-Manufacturing Technology. 2009 Dec 31;58(1):157-60.

[15] Chan LK, Wu ML. Quality function deployment: A literature review. European Journal of Operational Research. 2002 Dec 16;143(3):463-97.

[16] Saaty TL. What is the analytic hierarchy process?. InMathematical models for decision support 1988 (pp. 109-121). Springer Berlin Heidelberg.

[17] Durham WH. Coevolution: Genes, culture, and human diversity. Stanford University Press; 1991.

[18] D'Hondt T, De Volder K, Mens K, Wuyts R. Co-evolution of object-oriented software design and implementation. InSoftware Architectures and Component Technology 2002 (pp. 207-224). Springer US.

[19] Dorst K, Cross N. Creativity in the design process: co-evolution of problem–solution. Design studies. 2001 Sep 30;22(5):425-37.

[20] Maher ML, Poon J, Boulanger S. Formalising design exploration as co-evolution. InAdvances in formal design methods for CAD 1996 (pp. 3-30). Springer US.

[21] Cox BJ. Object-oriented programming: an evolutionary approach.

[22] Arnold K, Gosling J, Holmes D, Holmes D. The Java programming language. Reading: Addison-wesley; 2000 Jun 15.

[23] Yilmaz S, Seifert CM. Creativity through design heuristics: A case study of expert product design. Design Studies. 2011 Jul 31;32(4):384-415.