Human-centric product conceptualization using a design space framework

Yih Tng Chong *, Chun-Hsien Chen, Kah Fai Leong
School of Mechanical and Aerospace Engineering, Nanyang Technological University, North Spine (N3), Level 2, 50 Nanyang Avenue, Singapore 639798, Singapore

Abstract

Designing products to fit human needs, preferences and capabilities is an essential key to competitiveness. In contested product markets, the management of user-related knowledge is therefore critical. Studies have shown that the identification and implementation of user requirements are significant issues for determining successful product development, especially during the conceptual design phase. User requirements represented in a single or limited level of abstraction is inadequate for effective incorporation into the conceptual design process. Such representation of user needs is argued here to be associated with issues such as the errors of problem framing, which is a cause of inadmissible, uncreative or sub-optimal designs. In this paper, a human-centric knowledge organization structure, Design Space Framework, is established to facilitate the consistent incorporation of user information into the length of the product conceptualization process. The role of this structure in human-centric design is illustrated in a case study.

© 2008 Elsevier Ltd. All rights reserved.

1. Introduction

Need is the mother of all inventions. Succinctly represented by this expression is the principle of a class of design methodologies that regards the needs of human as the main seed of design. This design paradigm advocates the derivation of design specifications from a formalized set of customer requirements, and the applications were known to have improved enterprises’ market positions [1]. Design methodologies under this paradigm include the well-known quality function deployment (QFD) [2], and other related methods [13–5]. Chen et al. [6] reviewed comprehensively the state-of-the-art patents of customer-oriented design technologies and a number of them were based on the framework of QFD. These patents were filed by corporations including IBM [7], Lockheed Martin [8] and Kabushiki Kaisha Toshiba [9]. Design methodologies and patented technologies that advocate the philosophy of human-centered design have today formed an important cluster in design-related literature.

This paper describes the current interests in human-centered design pertaining specifically to the ‘front-end’ of the design process, i.e., the conceptual design phase. Conceptual design can be perceived as the acts of handling problems and solutions at high levels of abstraction. This design phase determines about 80% of products’ lifecycle cost, and has therefore been widely regarded as the most vital phase of design [10]. It has however been noted that existing computer-aided design (CAD) systems have instead focused on modeling physical products geometrically using data available only at detail design stage, and have largely ignored the support required during the conceptual design phase [11,12]. This developmental trend of CAD systems is therefore somewhat inconsistent with the needs of design applications.

Effective collaboration is often absent at this ‘fuzzy front-end’ [10] of design. This could be attributed to the supposition that conceptual design process is highly dependent on designers’ tacit knowledge [13]. Explication of knowledge and process formalization can thus be keys to support coordination in a cross-functional collaborative design environment [14]. Studies in this aspect possess the prospects of leading product developers to better engage the fortes of computer (e.g., vast storage, high computational power and connectivity) to assist them in this important phase of design.

2. Issues of human-centered product conceptualization

Studies have recognized that the identification and implementation of customer requirements are two significant issues to ensure successful product development [15]. These issues should be of greater significance during the conceptual design phase, given that considerable developmental cost would be locked during this design stage. Under the human-centered design paradigm, customers provide key source of input to the conceptualization process by voicing their needs, problems, concerns and preferences. The result of this design stage is a set of specifications [3] that provides the foundation for the framework of the product. One of the well-known tools that can support human-centered product conceptualization is the QFD [2]. QFD maps the relationships between the domains of customer needs, engineering characteristics and so on, and thereby aiding conceptual design tasks based on systematic references to the customer requirements. This design methodology...
has been the basis to many patents [7–9] and research papers [13,16–18]. In these publications, QFD was employed as the knowledge framework underpinning the systems. Though QFD is attributed with virtues such as its intuitive and parsimony nature, it is however not without shortcomings. Systems that were built upon on the framework of QFD could inherit the inadequacies of the QFD method.

Traditional QFD [2] prescribes the mapping of an abstract layer of customer attributes or customer requirements to the engineering characteristics. Like most types of knowledge, however, customer requirements can exist in multiple levels of abstraction. The representation of customer requirements in limited abstraction depth can lead to design solutions of inadequate details. Secondly, it can lead to errors in problem framing, which is a cause of sub-optimal design. As the absence of more detailed forms of customer requirements leads to indeterminacy of better detailed solution, the first consequence of the depth limitation is clear. For instance, Naes and Nyvold [19] prescribed a systematic concept exploration method that could handle a wide range of possible alternatives, supported by the voice of customers (VOC) during prototype testing sessions. The conceptual design process or the exploration of design concepts worked only with a single level of abstraction. The method takes care of the breadth of design alternatives; it is however not able to facilitate the process of concept detailing. Eagan et al. [20] summarized: framing is the most critical part of the process of decision making, because making the right decision in the wrong decision frame can be a serious error. The second consequence of treating customer requirements in a single (or limited) abstraction level is that the design problem could be over or under-framed. Proper framing is the key to work on the right problem. In the context of QFD where a set of customer requirements is mapped to a set of engineering characteristics, the latter set of information constitutes the frame of the given design problem. This problem frame acts to constrain the possible design solutions in a delimited design space. In other contexts, such as Ulrich and Eppinger’s method [3] and function-means map [21], the problem frames are the defined set of specification metric and design principles, respectively.

As the concept of problem frame is correlated to the user needs in human-centered design methodologies (e.g., in the context of QFD, engineering characteristics are related to customer requirements), the abstraction level(s) of customer requirements under design consideration can determine whether the design problem is aptly framed, over-framed or under-framed. In this vein, Al-Salka et al. [22] suggested that solution-neutrality is ensured by careful abstraction. In the case of over-framing, which is constituted by the acceptance of undue pre-constraints, the solution space will be unduly reduced. In such cases, the generation of creative concepts can be hindered. This phenomenon has been discussed in other literature by Eagan et al. [20], Pahl and Beitz [23], and Sim and Duffy [24]. On the other end, under-framing may lead to the generation of inadmissible design concepts. In both cases of framing errors, sub-optimal designs can be resulted. The task of problem framing is parallel to justifying and refuting pre-conceived solutions for a given problem. While Eagan et al. [20] generalized this issue of strategic problem framing in a holistic manner, their strategy has however not been further established as a design methodology.

In the context of product conceptualization, the risk of over and under-framing a design problem is associated with design methodologies that adopt only a single or limited range of abstraction levels in representing customer requirements. This risk can be mitigated with a knowledge framework that holistically treats the requirements in a wide spectrum of abstraction levels. Such knowledge structure would allow the problem frame setting to be flexible in the typical large space of design, and can therefore facilitate the dynamic adjustments of the frame during the process of conceptual design. Through such an approach, the risk of framing error can be adequately addressed.

Voices of customers should not only be elicited at the front-end of the conceptual design process, but rather frequently at various junctures along the process. Besides, not all requirements can be known at the outset of a design task [25]. It is therefore necessary for the designers to collect customer opinions consistently during the conceptualization phase, and undertake to incorporate them.

In view of the challenges mentioned above, a unique knowledge representation structure, termed Design Space Framework (DSF) is proposed to organize customer requirements in multiple levels of abstraction. In effect, conceptual solution exploration in depth and generation of creative design concepts can be facilitated. The framework also enables the systematic incorporation of the spectrum of user requirement abstractions into the process of product conceptualization.

3. The elements of conceptual design space

This section presents a set of objects that formalizes the knowledge acquired, operated and evolved in the conceptual design phase. These objects constitute a taxonomic scheme, which are primitives to a knowledge organization structure – Design Space Framework (DSF). Friedman perceived theory as an ordered set of assertions about a generic behavior or structure assumed to hold throughout a significant broad range of specific instances [26]. As depicted in Fig. 1, the theoretical postulations here serve as a filter that organizes the phenomena of the real-world as ordered events.

Instances of information conversed during conceptual design process can include: ‘this shaft transmits energy to the flywheel’, ‘an l-beam is between the blocks’, ‘this product has to weigh under three kilograms’, ‘the springs absorb the shock’, and ‘customers want to feel comfortable’, etc. The generation, operation and evolution of such information, including the voices of customers, are the dynamic processes within the conceptual design phase. To study the incorporation of customer requirements into product conceptualization process, complete forms of knowledge existing in the process such as those given in the above instances have to be modeled. In this work, the relevant information is generalized and orthogonally classified into finite sets of objects. In this vein, Definition 1 initializes the conceptual design domain knowledge of this model (as depicted in Fig. 2).

Definition 1. Conceptual design knowledge exists in a universe of discourse, domain $D$. 

![Fig. 1. Modeling real-world phenomena through theories.](image-url)
The taxonomy of domain $D$ and its inter-relationships are respectively defined in Sections 3.1 and 3.2.

3.1. Knowledge objects

Knowledge instances involved in the conceptual design phase, as seen in Fig. 2, are seemingly disorganized. To this end, this study postulates that they are definable based on structured objects in generalizable patterns. Consequently, knowledge of conceptual design can be discretized and generalized into orthogonal classes of information. An ordered assembly of the information classes, which is further discussed in Section 4, can therefore be employed to study and to direct the assimilation of user needs into the process of conceptual design. The classes of design information captured in $D$ are formalized in the rest of this sub-section. Prior to that, the set of finite knowledge objects is defined in Definition 2.

**Definition 2.** Let $N$ be the set of finite knowledge objects considered in $D$.

It was reported that Japanese firms have had overall product development processes that hinge on clear sets of customer-driven requirements [27]. In that respect, user needs are perceived as a vital class of information in design [28]. This class of information can be derived, possibly using various market research techniques such as focus group and interviews. Under various design methodologies [2–4], user need information is used to determine the design specifications of the product. In this model, it is assumed that every aspect of a final product can be duly traced back to the voices of customers. Whether it is a rounded edge or a choice of color, they can be reversibly traced to the related roots — customer requirements. The term ‘customer’ in this work does not exclusively refer to the end-users. Instead, it generally encompasses all stakeholders of the product lifecycle, including the end-users, the product managers and the production engineers for instances. **Definition 3** states the representation of the customer requirements in this model.

**Definition 3.** Let the customer requirements be represented by $c_i \in C$.

$$C = \{c_i | i = 1, \ldots, I\} \quad \forall c_i \in C \subseteq N$$

(3.1)

where $i$ is the sequential order for element $c_i$, and $I$ is the total number of element $c_i$ in a design problem.

Peeples and Boulton emphasized that incomplete or incorrect product requirements nearly guarantee a non-competitive product offering [27]. Product requirements are the second of the three general classes of information proposed to exist in the conceptual design space $D$. Other similar representations used in literature have terminologies such as functional requirements [4] and engineering characteristics [2]. A product requirement is defined, in this work, as an asserted product characteristic that can lead to the satisfaction of one or more customer needs. It is also the premise to design solutions. For instance, a designer noted that the toddler chair he is designing ought to be free from sharp edges (i.e., a product requirement) so as to ensure the safety of the end-users (i.e., a customer requirement). This product requirement is one of the information that he had prior to the generation of solutions. Product requirements in practice could be explicitly documented in certain format, or they can be tacit in the minds of the designers [13]. In the latter case, the information would be implicit to the rest of the project stakeholders. In this model, they are explicitly formalized as stated in **Definition 4**.

**Definition 4.** Let the product requirements be represented by $p_j \in P$.

$$P = \{p_j | j = 1, \ldots, J\} \quad \forall p_j \in P \subseteq N$$

(3.2)

where $j$ is the sequential order for element $p_j$, and $J$ is the total number of element $p_j$ in a design problem.

Design solutions can be characterized by design parameters [4] and schemes [5] amongst other types of representation terminologies. Solutions to design are regarded as the last of the three general classes of information in this work. Unlike the former two classes ($C$ and $P$), solutions may have alternatives in the context of product design. Alternative solutions, if any, therefore have to be inclusively represented in the framework of the proposing model. It should be noted that within a set of solution alternatives, only one is valid for the overall solution. As such, the representation of conceptual design solutions of a given design problem is mathematically stated in **Definition 5**.

**Definition 5.** Let the design solutions be represented by $s_k^l \in S$.

$$S = \{s_k^l | k = 1, \ldots, K \wedge l = 1, \ldots, L_k\} \quad \forall s_k^l \in S \subseteq N$$

(3.3)

where $k$ is the sequential order for element $s_k$; $K$ is the total number of element $s_k$ in a design problem; $l$ is the sequential order for alternative solutions of $s_k^l$; and $L_k$ is the total number of alternative solutions of $s_k$.

Any complete single product can be perceived as a form of solution. For instance, a mug is a solution to hold beverage. In the context of this work, such subject of design task is treated as a solution — the most abstract one — to a given problem. A special element of $s_k^l \in S$ where $k = 1 = 0$ (i.e., $s_k^l \in S$) is in this model set aside to denote the general solution of a given design problem. Take for instance, in the context of solving a fluid backflow problem, the designers have been tasked to design a check valve; in this case, the solution of the highest abstraction level (denoted by $s_0^0$) could simply be labeled as ‘check valve’.

Based on **Definitions 3–5**, Eq. (3.4) states that object classes $C$, $P$ and $S$ are the three subsets of set $N$.

$$N = \{C, P, S\}$$

(3.4)

3.2. Inter-relationships of knowledge objects

As asserted by Nishioka et al. [29], knowledge is characterized by the connections amongst them. The set of relationships amongst the three general types of information classes defined in Section 3.1, viz. customer requirement ($C$), product requirement ($P$) and solution ($S$), are proposed and formalized in this section.

**Definition 6.** Let $E$ be the set of finite design relationships amongst the knowledge objects considered in $D$, the conceptual design knowledge domain.

In various design methodologies [3,4], direct relationships exist between the represented customer requirements and product requirements as well as between the product requirements and
design solutions. These two classes of relationships are maintained in this work, and are described in Definitions 7 and 8, respectively.

**Definition 7.** Let the relationships between objects \( C \) and \( P \) be represented by \( u_{ij} \in U \).

\[
U = \{ u_{ij} | i \in \{ 1, \ldots, I \}, j \in \{ 1, \ldots, J \} \} \quad \forall u_{ij} \in U \subseteq E \tag{3.5}
\]

**Remark 1.** Eq. (3.5) reflects that \( \forall p_j \in P \) has a corresponding parent, \( c_i \in C \), related by element \( u_{ij} \in U \).

**Definition 8.** Let the relationships between objects \( P \) and \( S \) be represented by \( v_{lk} \in V \).

\[
V = \{ v_{lk} | i \in \{ 1, \ldots, J \}, k \in \{ 1, \ldots, K \}, l \in \{ 1, \ldots, L_k \} \} \quad \forall v_{lk} \in V \subseteq E \tag{3.6}
\]

**Remark 2.** Eq. (3.6) reflects that \( s^c_l \in S \), other than \( s^p_k \in S \) as explicitly excluded, each has a corresponding parent \( p_j \in P \), related by element \( v_{lk} \in V \).

Fig. 3 schematically depicts these two sets of relationship. The solution to one problem becomes the basis of the next [30]. The logic asserted in this statement supports the existence of another set of relationships apart from \( U \) and \( V \) in domain \( D \). In the context of real-life problem solving, every solution is simultaneously a problem, possibly awaiting further solutions. For instance, the decision to buy a car may solve a traveling problem, but will remain as a problem, possibly awaiting further solutions. Take for instance, the solution to the initial given problem. To this end, an inquiry may arise: “Does this cyclical process terminate? And if it does, when?” An answer to the inquiry can be: “Only when a problem-solver needs to clarify the details of a solution, one would then consider the solution as a problem; otherwise, the recurring problem-solution pattern would terminate.” In the context of conceptual design, this logic is similarly true and applicable. The result of the above self-evident assertion is written as **Definition 9**. This definition is an essential link in the proposition presented in this paper.

**Definition 9.** A design solution is a design problem if and only if further detail of the solution is required.

Common to various design methodologies, the step after problem definition is to elicit for customer requirements, based on the defined problem [3]. This procedure suggests that a defined problem can be the premise for object \( C \) (Definition 3). Since **Definition 9** supports the equation of design problem to design solution \( s \in S \), it can be inferred that object \( S \) (design solution) can be the premise of object \( C \) (customer requirement), under the condition stated in **Definition 9**. Based on such inferences, causal relationship exists between \( S \) and \( C \) in this model, in the manner expressed in **Definition 10**.

**Definition 10.** Let the relationships between domain \( S \) and \( C \) be represented by \( w_{kl} \in W \).

\[
W = \{ w_{kl} | k \in \{ 1, \ldots, K \} \land \exists l \in \{ I, \ldots, L_k \} \land l = 0, \forall i \in \{ 1, \ldots, I \} \} \\
\forall w_{kl} \in W \subseteq E \tag{3.7}
\]

**Remark 3.** Eq. (3.7) reflects that \( \forall c_i \in C \) has a corresponding parent \( s^c_l \in S \) (including \( s^p_k \in S \)), related by element \( v_{lk} \in V \). The equation does not however state that all \( s^c_l \in S \) has a child \( c_i \in C \). This means that \( s^c_l \in S \) can be a terminal node (in the terminologies of graph theory). Such assertion in the context of product design conforms to the fact that not all design concepts are necessarily further developed. For instance, the development of less favoured alternatives and non-qualified solutions could be terminated, thus bearing no further details.

Eq. (3.8) can be inferred from Definitions 7–9.

\[
E = \{ U, V, W \} \tag{3.8}
\]

Definitions 2 and 6 with Eqs. (3.4) and (3.8) jointly form the theoretical basis (as illustrated in Fig. 1) that model the conceptual design knowledge domain \( D \), as initially defined in **Definition 1**. Fig. 4 illustrates the classes and sub-classes of the objects in domain \( D \).

In this work, **Definition 9** essentially facilitates the modeling of the conceptual design development—a cyclical process of design knowledge generation, progressing towards the emergence of design concepts. This process model of conceptual design is illustrated in Fig. 5. The role of customer requirements in product conceptualization, the key interest of this paper, is schematically illustrated in Fig. 5.

**4. The definition of design space framework**

Two modes of operation between sketches in the process of conceptual development were identified [14,31]. They are vertical and lateral transformations. For series of sketches that are progressive in details, they are said to be transforming vertically into the depth of abstraction levels. On the other hand, lateral transformations bring sketches from one idea to another, i.e., the generation of alternative designs. These sketching patterns during product conceptualization reflect the cognitive behaviour of the designers. This
Conjecture permits the notion that product conceptualization is the act of exploring design space, vertically and laterally. On such basis, this work proposes two dimensions – vertical and lateral – to characterize the defined conceptual design space, $D$ (Definition 1). In this work another dimension in $D$ is deemed necessary to describe design concepts; since design solution can be characterized by its components, a third dimension is postulated in $D$ to accommodate the divisions of objects.

*Definition 11.* The three dimensions of space $D$ are $X$, $Y$ and $Z$, where

- **$X$:** Sibling objects along dimension $X$ are the decomposed constituents of their parents, having AND relationships amongst them.
- **$Y$:** Sibling objects along dimension $Y$ are alternative children of the parent, where only one is relevant to the parent for any instance of complete solution. They hence have OR relationships amongst them.
- **$Z$:** Objects generated along dimension $Z$ are of varying abstraction.

Three sets of formalism were established: (1) the three dimensions of $D$ (Definition 11), (2) the objects in $D$ (Definition 2), and (3) the objects’ inter-relationships (Definition 6). Based on these, objects in a definitive order in space $D$ can be generalized as a knowledge organizational structure – Design Space Framework (DSF). It is postulated that DSF can serve as a knowledge organization structure to model design concepts (Definition 12), and its instantiation as a specific design concept can be mathematically represented (Definition 13).

*Definition 12.* Let Design Space Framework be a knowledge organization structure for product conceptualization.

*Definition 13.* Let an instantiated Design Space Framework be an AND/OR tree graph $G$, a quadruple where

$$G = (N, E, \Phi, \Gamma)$$

and the functions $\Phi$ and $\Gamma$ are the forward and backward pointers, respectively.

A DSF, or specifically Graph $G$, has been proposed to exist in space $D$, which can be characterized by the said three dimensions. Any particular design concept can be modeled by $G$ – an assembly of the elements in sets $N$ and $E$ (see Fig. 6). This representation is a type of network model, where the nodes represent objects and the arcs represent the associations or the relationships between them.

Dimension $Y$ defined in Definition 11 facilitates the representation of alternative solutions of any sub-component at any abstraction level of the overall design solution. To model and present the final design solution, dimension $Y$ can be suppressed. All unselected alternative solutions that are irrelevant to final solution can be isolated behind the $Z$–$X$ plane, along the $Y$ dimension. Consequently, the final conceptual solution of a product can be fully represented and simply presented on the two dimensional $Z$–$X$ plane.

In the context of artificial intelligence, tree graphs have been useful for knowledge representation and operation [19]. Tree graphs hold two conditions: (1) root exists, and (2) there is no cycle in the graph. In graph $G$, a special case solution node $s_0 \in S$ denotes the solution of the highest abstraction level within a given design problem. Remarks 1–3 state that every node in set $N$ has one parent.
each, with the exception for element $s_s^c \in S$ (Remark 2). In graph $G$, $s_s^c \in S$ is therefore the unique node without any parent. And all paths in the graph have to originate from it given that every other node has exactly one parent. As root exists in graph $G$, the above first condition is satisfied. The fact that all nodes (other than the root) each have one and only one parent indicates that no path can contain any node more than once. Therefore looping or cycling must not be present in graph $G$. Graph $G$ could thus serve as a data model that can be operated upon as a tree graph in the context of numerous artificial intelligence methodologies [32]. DSF as a tree graph can also be used to effectively describe product concepts and the rationales of their existences.

5. An illustrative case study

To illustrate the role of DSF in the conceptual development process, a case study of disposable coffee cup design is presented. It exemplifies how DSF facilitates the incorporation of customer requirements of various abstraction levels into the junctions of the process. As the result, the detailing of the design concepts is supported, and the risk of under or over-framing of the design problem is greatly reduced. The case study also shows that DSF can serve to direct product developers in the elicitation of customer requirements. This case assumes that a design firm is tasked with designing a disposable coffee cup. The design firm elicited several design requirements from the client, viz. the product has to be stylish, environmentally friendly, portable and able to keep the beverage warm.

The team of designers embarked on conceptualizing the coffee cup – codenamed Auburn. Let Auburn be $s_s^a$, i.e., the solution for the problem posed by the client. Treating $s_s^a$ as a problem (with respect to Definition 9), the first set of customer requirements was formalized: $c_1$ – stylish image, $c_2$ – environmentally friendly, $c_3$ – ability to keep beverage warm, and $c_4$ – portable. Based on these customer requirements, the designers generated a set of characteristics that the product shall possess. The specified product requirements were: $p_1$ – elegant shape, $p_2$ – beautiful motif, $p_3$ – appropriate color scheme, $p_4$ – readily recyclable material, $p_5$ – insulative design, $p_6$ – heat addition function, $p_7$ – no spillage during transportation, and $p_8$ – easy to carry. For each of the identified product requirement, the design team explored a range of conceivable solutions. The relationships between the above mentioned customer requirements, product requirements and the generated solutions are shown in Fig. 7. Each plausible solution was articulated and sketched. With respect to $p_1$ (elegant shape), four alternative solutions $s_1$, $s_2^3$, $s_1^4$ and $s_2^4$ were conceived, as illustrated in Fig. 8. Accordingly, three alternative solutions to $p_2$ (beautiful motif) were conceptualized. They are as shown in Fig. 9.

To meet the product requirement $p_4$ (readily recyclable material), the design team jotted several recyclable materials that are deemed suitable for manufacturing disposable coffee cups. They were: $s_4^1$ – polypropylene, $s_2^1$ – expanded polystyrene, $s_1^2$ – polyethylene terephthalate, and $s_2^2$ – certain food-grade paper. It was noted that solution $s_2^2$ was relatively abstract. Indistinct information in conceptual design stage is known to be common [3], and should be tolerated. In this case, $s_2^2$ may be allowed to remain abstract, and can be further concretized in the later stage.

To minimize heat loss from the cup, the designers proposed several ideas as follows: $s_3^1$ – polystyrene as cup material, $s_2^3$ – polystyrene as cup sleeve, and $s_3^4$ – cardboard as cup sleeve. The design team asserted that other than having the usual insulative characteristics of coffee cups, having heat addition function would be an innovation to sustain the high temperature of the beverage – only if it is technically and economically viable. Such an innovative idea was derived from the product requirement $p_6$ (heat addition function). Chemical heat source ($s_4^3$) was then identified as a solution. The designers vaguely knew that the release of an activating liquid solution into contact with certain form of metallic compound [33] would produce enough heat to warm up a cup of coffee. Based on this vague idea, the team drew up a conceptual sketch, as shown in Fig. 10. As the finer details of this concept requires specialized domain knowledge that the design team is deficient in, this sub-solution can then be out-sourced to a vendor for further conceptualization.

To satisfy product requirement $p_7$ (no spillage during transportation), the designers deemed that a cup lid ($s_4^4$) could be the solution. As for product requirement $p_8$ (easy to carry), the team generated two alternative solutions – either a carrier tray ($s_4^2$) or an attachable handler ($s_4^3$). The designers proceeded to pursue the details of the respective alternatives, before comparing and making a choice between them.
With respect to Definition 9, carrier tray (s2) and attachable handler (s1) were perceived as design problems. At this juncture of the design process, the designers need to gather client’s comments that are specific to the two solutions or problems. The client stated that any handler of the cup has to be aesthetically designed, and removable. Based on these elicited requirements of the handler, two child nodes were appended to node s1: c5 – aesthetic, and c6 – modular (see relationships in Fig. 7). Accordingly, the relevant product requirements were generated, followed by the respective solutions. The conceptual sketch of the resulted attachable handler is presented in Fig. 11. The designers proceeded to generate the details of the carrier tray design before selecting one of these two solutions to satisfy the product requirement p6. This short design case outlines the structure of DSF. The concept of Auburn took the form of graph G, which captured the design outcomes and also the rationales of their existences (the origin of design solutions can be traced, and unselected design alternatives are documented). To design a coffee cup is simple, relative to other sophisticated engineering designs. If DSF were to be applied on a more complicated product, wider and deeper graph G can be expected to model the design concepts.

6. Discussions

The identification and implementation of customer requirements are significant issues for successful product development [15], especially so during the conceptual design phase. Methods that facilitate the incorporation of requirements during product conceptualization include QFD and the product development procedures prescribed by Ulrich and Eppinger [3]. These methods do not however address the user needs in a spectrum of abstraction levels.

An advantage of representing customer requirements in multiple abstraction levels, as in the DSF, is that each level can be mapped to a corresponding level of the design solution. In the case illustrated, the lower-leveled requirement of modularity (c6) directs the design of the disposable coffee cup at a more detailed level, i.e., the design of the cup holder. References can thus be made to the multi-leveled customer requirements in DSF when the designers generate solutions at different level of detailness. Further, generating concepts in several cycles or steps of abstraction is less complex than generating solution with all details included in a single-step. Methods that assist conceptual design, such as the QFD [2], prescribe single-step solution generation (i.e., single abstraction level solution representation). Such method may be difficult to apply on more complicated products. Leveraging on Definition 9, the proposed structure of DSF captures conceptual design progression as a problem-solution cyclical process. The result is the division of design information into multiple abstraction levels, and therefore facilitates the representation and handling of more complex products.

The row of technical characteristics in the QFD matrix is mapped to fulfill the column of customer attributes. Design solutions are then generated from those specifications, which forms the frame of the design problem. If the problem frame is not set at a suitable abstraction level (i.e., over-framed or under-framed), an uncreative, inadmissible or sub-optimal solution can be resulted. This risk is mitigated in the structure of DSF. Unlike in QFD, the frame of problems in DSF is flexible. In the example of the case discussed, it is shown that DSF does not constraint the design solution to have an attachable handler (s1), even if that is included in the initial problem statement given by the client. DSF facilitates the designers to navigate along the dimension of abstraction, to rethink assumptions and to challenge unnecessary pre-conceived solutions; the problem frame of conceiving a attachable handler (s1) can possibly be dynamically adjusted during the process of product conceptualization, to focus on the more abstract (higher-leveled) requirement of portability (p6), or to shift to an alternative problem frame of a carrier tray (s2) design. DSF therefore prevents the risk of designers having fixation on any problem frame, which can lead to compromised design solution.

Engelbrektsson and Soderman [15] identified the problem of customer requirements missed or discovered untimely, leading to costly consequences. They studied this issue in terms of techniques used to elicit customer voices (such as questionnaires and clinics), and concept representation (such as sketches or mock-ups). Apart from considering the methods of elicitation, how the elicited customer requirements get internalized into the design process is not less critical. A design methodology that systematically incorporates customer requirements along the process of conceptual development can be useful to implement and streamline the process, gaining the positive effects of the human-centered philosophy.

Voices of customers should not only be elicited in the front-end of the conceptual design process, but frequently at various junctures along the product. Based on today’s spiral and stage-gate processes of design, inputs from customers should be iteratively elicited during the product development process [34]. At various levels of product concept details, customer voices are required to guide and support decisions for human-oriented design. A rationale that advocates the consistent collection and incorporation of customer opinions during the product design process is the fact that not all requirements can be known at the outset of a design task, as identified by Gero and Kannengiesser [25]. In other words, certain customer requirements cannot possibly be elicited without the intermediating work of the designers. This phenomenon is exemplified in the presented case study. The client of project Auburn was not likely to specify the requirements of the attachable handler at the outset, until the cue given by the designers that it could be one of the possible sub-solutions. This feature of conceptual design phase further necessitates the collection of customer opinions consistently along the conceptualization phase.

In the context of product design informatics system, the manner of representing and associating design information is directly related to the efficacy of the systems. Apart from the outcome of the conceptual design phase, product developers also should
possess the information on the way the final design was arrived, including ‘why’ and ‘why not’ for sub-solutions at all abstraction levels. DSF not only represent the adopted solutions in a product, it also captures unselected design alternatives at various levels of abstraction (as seen in the case study). In DSF, sub-solutions of all levels would have customer requirements as their origin. As such, every feature of the design solution can be duly traced back to the customer requirements. DSF therefore, in the above manner, codifies the finalized design concept and captures the rationales of their existences. The instantiated graph $G$ resulted from the product conceptualization phase can be employed as the basis for downstream design; the data structure of DSF may be mapped to a conventional CAD software, for the continuation of the product development process. Such integration of systems would allow the tracking and retrieval of early design rationales.

The structure of the proposed DSF reflects and facilitates the practical needs of systematic customer requirements incorporation into the conceptualization process. By having $ci$ nodes acting as the essential bridges intermediately along the product conceptualization process (i.e., along dimension $Z$), customer requirements can be specifically elicited with respect to the respective sub-problems $(s^i)$ of the abstraction levels. This structure ensures that relevant customer opinions are constantly sought and systematically applied in the entire product conceptualization process. In effect, this knowledge structure advocates human-centered design philosophy by establishing the correlations between customer voices and product attributes at multiple abstraction levels of the solution. From another perspective, DSF can be applied as a road map or basis for customer voice elicitation.

7. Conclusions

The incorporation of customer requirements into product conceptualization process is critical under the human-oriented design paradigm. This paper addresses this issue by postulating a knowledge organization structure – Design Space Framework. It is known that the intensity of knowledge refers to not only the amount in a chunk, but more importantly, how well systemized is the knowledge in the chunk [35]. DSF codifies the tacit knowledge dealt with in the conceptual design process based on an assembly of representation techniques. In this framework, the taxonomy has by no means represented the depth of information required in a knowledge intensive environment. The structure however offers a platform for advanced developments. For instance, the domain-neutral DSF can be further defined to characterize particular classes of products, to support the handling of specific ranges of conceptual design problems. This work postulated a structure that provides a methodical basis to incorporate customer requirements. Advanced application and implementation of the technique can be further achieved through specifications of computational objects such as lower-leveled data structures and algorithms.

References