

Managing Functional Coupling Sequence to Decrease Complexity and Increase Modularity in Conceptual Design

Chu-Yi Wang^{1,*}, and Stephen C.-Y. Lu¹

¹Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90089-1453, USA

Abstract. Conceptual design is the stage where the upstream objectives and downstream constraints meet, so both the ideality and practicality are important in conceptual design. Modularity supports designing complex system while the design can take advantages from the downstream resources. According to Axiomatic Design Theory (ADT), the design with the least complexity is the ideal design, and managing functional coupling sequence can lead to the minimal complexity of design concepts. In this paper, Design Coupling Sequence (DCS) was introduced to bridge the ADT and the modular design. The ‘precedence’ and ‘functional sets’ were defined in DCS to manage the coupled design concepts in order to support the modularity of the design concepts. The ‘precedence’ identified by the level of functional coupling helps realize the sequencing order. The two ‘functional sets’ were defined as the independent U-set: the collection of functionally dependent concepts, and the coupled C-set: the collection of the strongly coupled concepts. The faucet design case study shows how the DCS method and its strategy to decrease relative complexity and increase modularity, and the results between design by extra coupling elimination and design by the existing modules are compared. The DCS method bridges the design theory to design practice.

1 Introduction

Conceptual design is an early design stage that is commonly regarded as the transformation stage from Functional Requirement (FR) to Design Parameter (DP) [1-5]. Since the conceptual design is where upstream objectives (i.e., Customer Needs, CNs) and downstream constraints (i.e., Process Variables, PVs) are “balanced” through creative mappings (i.e., ideations) from FRs to DPs, it is important for innovators to ideate design concepts that are as ideal and practical as possible. The “ideality” of a product refers to its ability to use the most superlative manner to satisfy upstream market demands during product development, and in this paper it follows the definition in Axiomatic Design Theory (ADT) [6], that the ideal design is the simplest design (i.e. the design with minimal complexity). The “practicality” of a product refers to its ability to take advantage of existing resources at downstream of product development. To approach ideal design, functional independence is essential to minimize the relative complexity; to approach practical design, physical integration is favoured to take advantages of the downstream resources or database, which often are modules and existing components (i.e. to increase modularity).

Some case studies [7-9] have shown the usefulness of using ADT to reduce the complexity in conceptual design. However, Chen et al. [10] discussed the difficulties of directly using ADT to modularize the design concepts. The gap between design theories and design practicality needs to be bridged. Wang et al.

proposed Design Coupling Sequence (DCS) method to decrease the complexity [11]. In this paper, the DCS method is further modified and explained for not only complexity reduction but also modularity enhancement.

1.1 Complexity Reduction by Axiomatic Design

1.1.1 Real Complexity and Imaginary Complexity

Logically, the concept generation process from FR to DP follows synthetic proposition [12] so that the relationship between FRs and DPs is built. In the ideal case, each of the FRs “is satisfied by” only one DP. However, in the most design cases, one FR is satisfied by two or more DPs because it is difficult to follow the Independence Axiom all the way, so that the functional coupling exists in the design. According to ADT, the functional coupling relationship between FR and DP can be depicted by design matrix (DM) where X denotes the non-negligible relationship and O denotes the negligible relationship. Since there is no specific value for each parameter in such early stage, X presents a placeholder for the non-negligible relationship. For example, the DM in Fig. 1 shows that FR₁ is satisfied by DP₁, DP₂, and DP₄; in other words, there is a functional coupling of FR₁ between DP₁, DP₂, and DP₄ in the case. In Suh’s Complexity Theory [6], relative complexity includes real and imaginary complexities. The real complexity is caused by the functional couplings between DPs, which are captured as the ‘X’s at the off-diagonal locations in

* Corresponding author: chuyiwan@usc.edu

the DM. When it is the ideal case, the DM is a diagonal matrix, which has no complexity (i.e. uncoupled design). Oppositely, if every FR is satisfied by all the DPs, then the design is a full design matrix, which has 100% complexity (i.e. fully coupled design). In the halfway, the design with a solid triangular matrix is a decoupled design. These three typical design types were introduced in Suh's Axiomatic Design Theory [13]. The rest design types are a triangular but not solid DM and a non-diagonal non-triangular DM, which are named partially decoupled design and partially coupled design respectively in this paper (Fig. 2).

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \end{bmatrix} = \begin{bmatrix} X & X & O & X \\ O & X & O & O \\ X & X & X & X \\ O & X & O & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \end{bmatrix}$$

Fig. 1. An example of the design matrix.

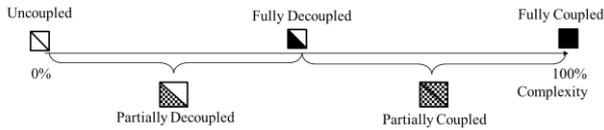


Fig. 2. The relationship between the complexity and design matrix

Imaginary complexity arises when the designer lacks understanding of the system. For the example in Fig. 1, the complexity of the design would be higher than fully decoupled design's because the DM is partially coupled, but if one understands the system, the DM could be rearranged into a fully decoupled design as shown in Fig. 3, then the complexity could be reduced. That extra complexity is imaginary complexity.

$$\begin{bmatrix} FR_2 \\ FR_4 \\ FR_1 \\ FR_3 \end{bmatrix} = \begin{bmatrix} X & O & O & O \\ X & X & O & O \\ X & X & X & O \\ X & X & X & X \end{bmatrix} \begin{bmatrix} DP_2 \\ DP_4 \\ DP_1 \\ DP_3 \end{bmatrix}$$

Fig. 3. The rearranged DM from Fig. 1.

The order of DP in Fig. 3 leads the design to the minimal real complexity, and the order is called "execution sequence" because the designer can have the minimal real complexity when he/she uses the sequence of DP to execute the concept decision-making during concept improvement. Therefore, in ADT, the imaginary complexity can be quantified by the equation 1, where z is number of the execution sequences, and $m!$ is the total number of sequences for the design with a m -by- m design matrix. According to the equation 1, the imaginary complexity can be reduced if one can know more execution sequence.

$$C_I = -\log_2 \left(\frac{z}{m!} \right) \quad (1)$$

1.1.2 Complexity Reduction with Design Coupling Sequence

Based on above, both real complexity and imaginary complexity can be reduced by extra coupling elimination, design matrix rearrangement, and/or execution sequence. As shown in Fig. 4, except uncoupled design and fully coupled design cases in which the execution sequence is trivial, execution sequence is the essential step during the concept improvement. Design Coupling Sequence (DCS) was proposed by Wang et al. [11] to obtain the execution sequence for partially coupled and partially decoupled designs. DCS uses set, branch, and arrow to reconstruct the functional coupling relationship into sequence, and proposed a concept of "precedence" to help the determination of the order of DP.

The steps of DCS in the paper include count Xs, list DPs, manage underlined DPs, and arrange into sequences. To count Xs aims to obtain the precedence. However, the precedence is not as simple as the non-zero term of the DP, the number of the non-zero term of the FR affects the precedence. Therefore, a revised algorithm is proposed in this paper to help obtain the execution sequence for complexity reduction.

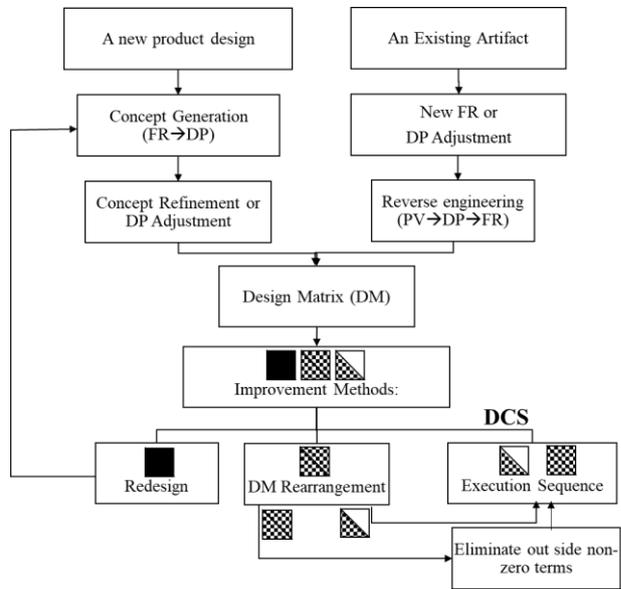


Fig. 4. The concept improvement roadmap.

1.2 Design Modularity

1.2.1 Modularity and Functional Coupling

Modularity is a bottom-up process that arises from the decomposition of a product into subassemblies and components [14]. In other words, it is some isolated collections/sets of the components in the system. During the modularity process, some metrics or methods are used to cluster the components. In conceptual design, the components are the DPs, and the metrics or the methods could be developed from the information of DM or Design Structure Matrix (DSM), where there depicts the relationship among the DPs.

Most modular designs more focus on the interaction of the DPs [15-18], so the DSM is usually used. However, clustering the DPs with DSM may result in additional coupling or unwanted imaginary complexity. For example, FR_1 is satisfied by DP_1 only, and FR_2 is satisfied by DP_2 and DP_3 , so DP_1 and DP_2 are functionally independent, and DP_2 and DP_3 are functionally coupled. If DP_1 and DP_2 are grouped as a module because of the physical integration, when improve DP_3 to better satisfy FR_2 , the module of DP_1 and DP_2 needs to be improved correspondingly, then the DP_1 is coupled with DP_3 because of the grouping. Since conceptual design stage should consider both upstream side and downstream side, DM would be better than DSM for modularity during conceptual design as it carries the coupling information between FRs and DPs.

1.2.2 Using DCS to Support Modular Design

It is difficult to direct use DM to identify the modules because the design is usually either in the partially coupled type (such as the example in Fig.1) or with the different amount of FR and DP (i.e. rectangular DM) in most design cases. DCS is a functional coupling sequence management method which can apply to rectangular design matrix and prescribe execution sequence for partially coupled design. In this paper, two types of functional sets to cluster DPs with the DCS results to increase the modularity is proposed. The research goal is to develop a theoretically sound practical approach that can reduce relative complexity on one hand and to increase modularity on the other hand during the conceptual design stage.

2 The Redefined Precedence and The Functional Sets

For obtaining execution sequence, the precedence in DCS includes higher, lower, and equal precedence. Previously, it was defined by the number of non-zero terms of the DP in DM. Since the number of the non-zero term of the FR impacts the functional coupling as well, the precedence is redefined as the elaboration in Section 2.1.

For increasing modularity, the two functional sets are defined to use DCS results to support modular design. They are the independent U-set: the collection of functionally dependent concepts, and the coupled C-set: the collection of the strongly coupled concepts. Because U-sets are mutually independent and C-sets find the DPs that need to be improved together, the designer can select concepts or get ideas from the existing modules according to the sets. The details are presented in Section 2.2.

2.1 Precedence

The precedence indicates the priority of the DP in order for less complexity. In other words, it determines which DP should execute first during concept improvement.

The rules were developed based on Linear Algebra and Suh's design matrix rearrangement method [19]. It has two rules:

- The DPs that relate to the FR which has the less sum of the correlation numbers in its row are in the higher level of precedence, and
- The more functionally coupled DPs (i.e. the larger sum of the numbers in its column) has the higher precedence.

Using Fig. 5 as an example, the related FRs to DP_1 and DP_3 have one non-zero term (for FR_1) and two non-zero terms (for FR_3) respectively, so DP_1 has higher precedence than DP_3 . The DP_2 has higher precedence than the DP_3 because the DP_2 has more non-zero terms than the DP_3 does.

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \\ FR_5 \\ FR_6 \\ FR_7 \\ FR_8 \\ FR_9 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & X & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & X & X & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & X & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & 0 & X & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X & X & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X & X & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \\ DP_5 \\ DP_6 \\ DP_7 \\ DP_8 \\ DP_9 \end{bmatrix}$$

Fig. 5. A sample design matrix to explain precedence.

When the above metrics are equal, these DPs have equal precedence. There are two types of equal precedence:

- 1) If these DPs are functionally independent, the order of these DPs doesn't matter (i.e. any of them can execute first).
- 2) If there are functional couplings between these DPs, they would be the unsolvable DPs (strongly coupled) and should be consider together.

For example, the DP_1 and DP_9 in Fig. 5 are with the type 1 of equal precedence. Also, the DP_5 and DP_6 in Fig. 5 are with the type 1. The DP_7 and DP_8 in Fig. 5 are with the type 2 of equal precedence.

2.2 Functional Sets

In this paper, two types of functional sets are defined based on the functional coupling and the equal precedence.

- 1) The completely independent set **U**: U-set is the collection of all functionally dependent DPs in the system, so that the set is independent to any other U-sets in the system. U stands for "uncoupled".
- 2) The insolvably coupled set **C**: C-set is the collection of these coupled concepts that can't be decoupled by sequencing, so these groups of DPs must be considered together as a module. C stands for coupled, and it is the type 2 of equal precedence.

In Fig. 6, there are five U-sets in this DM, and they are U_1 : DP_1 , U_2 : DP_2 and DP_3 , U_3 : DP_4 , DP_5 , and DP_6 , U_4 : DP_7 and DP_8 , and U_5 : DP_9 . There is only one C-set in the DM, which is the set of DP_7 and DP_8 . The set of DP_7 and

DP₈ is fully coupled as a C-set, but it is independent from other DPs in the system, so it is also a U-set.

$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \\ FR_5 \\ FR_6 \\ FR_7 \\ FR_8 \\ FR_9 \end{bmatrix}$	$=$	$\begin{matrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \\ U_7 \\ U_8 \\ U_9 \end{matrix}$	$\begin{bmatrix} X & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & X & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & X & X & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & X & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X & 0 & X & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X & X & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X & X & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & X \end{bmatrix}$	$\begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \\ DP_5 \\ DP_6 \\ DP_7 \\ DP_8 \\ DP_9 \end{bmatrix}$
--	-----	---	---	--

Fig. 6. A sample design matrix to explain functional sets.

3 The Modified Design Coupling Sequence Method

The modified DCS method uses the same representations and the same way to transform from DCS results to execution sequences. The differences from the original DCS are the algorithm to order the DPs and the view point and the use of the sets in DCS.

Following sections will give a short introduction of the representations and the execution sequence, the modified DCS algorithm, and the strategies of using DCS.

3.1. DCS Introduction

3.1.1 The Representation of DCS

Based on the section 2, a typical outcome of the DCS method is some U-sets of DPs containing some C-sets and/or arrows (Fig. 7). U-set in DCS refers to a functionally independent set that is completely uncoupled with other sets, and the arrow (\rightarrow) in the U-set shows the direction of the sequence. C-set is represented by braces $\{ , \}$, and it indicates the unsolvable couplings. Branch in the U-set is uncoupled with other branches, and it is a kind of the type 1 of the equal precedence.

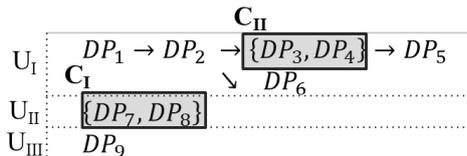


Fig. 7. An example of DCS outcome for representation explanation.

U-sets and C-sets function differently. An U-set is like a unit that can be an independent module, or can physically integrate with other U-sets. A C-set denotes the strongly coupled DPs, whose couplings can not be decoupled by sequence. The functional coupling between the DPs in a C-set is suggested to be eliminated,

or the designer would consider all the DPs in the set as a module while determining the design concepts.

The DCS algorithm orders the DPs by their “precedence,” and the arrow in the U-set indicates higher to lower precedence on one hand, and the sets and branches indicate equal precedence on the other hand.

3.1.2 The Execution Sequence

The execution sequences can be formed from DCS algorithm outcome by permutation and combination. Arrow indicates the order; besides, branch can be any order with other branches, and the order of the DPs in the branch follows above rules. The order in a C-set is meaningless because all the DPs in the set should be determined together. The number of all acceptable DCS sequences (that can minimize the complexity) is calculated by multiplying the factorial of the number of the U-sets and that of the branches for each node. The formula can be written as

$$u! \times \prod_{i=1}^n b_i! \quad (2)$$

where u is the number of the U-sets, b is the branch number of the node, and n is the number of the node.

For the example in Fig. 7, there should be 12 acceptable sequences in that design ($3! \times 2! = 12$).

3.2 DCS Algorithm

3.2.1 The Algorithm Logic

The logic of the algorithm firstly is following the precedence to order each DPs. When assess the level of the precedence, the level of the functional coupling with FRs is examined first then the level of the functional coupling of DPs. Meanwhile, the equal precedence DPs are labelled as groups to further examine. Next, each member in the group will be examined the functional coupling with its group members. Those equal precedence with functional coupling DPs indicates the unsolvable coupling, which is represented by the braces of C-set. Finally, the functional coupling relationship of each DP ordered by precedence is examined to build the connection forming the U-set.

3.2.2 The Notions in the Algorithm

Some notations are needed to better explain the algorithm. The calculation in the algorithm uses zero for 0 and one for X with the binary design matrix. The element at row “ m ” and column “ n ” in the DM is denoted by DM_{mn} . The sets

$$S_{DP_n}: \{m \mid DM_{mn} \neq 0\} \quad (3)$$

$$S_{FR_m}: \{n \mid DM_{mn} \neq 0\} \quad (4)$$

express the position of those non-zero elements which indicate functional couplings between FRs and DPs. In addition, the parameters

$$Sum_DP_n = \sum |DM_{mn}| \quad (5)$$

$$Sum_FR_m = \sum |DM_{mn}| \quad (6)$$

represent the degree of coupling of a specific DP and FR.

3.2.3 The Steps of the Algorithm

There are three steps in the modified DCS algorithm: put DPs in a line by precedence, manage equal precedence DPs, arrange DPs into U-sets.

Step 1: put DPs in a line by precedence

Find the S_FR_i that its Sum_FR_i is the smallest, and obtain the n in S_FR_i . Add the DP_n in a line, and then remove the column of the DP from the DM. Repeat this step until all the DPs are in a line.

If there are more than one different n , add these DP_n in the line by Sum_DP_n from the most to the least, and mark the DPs with same values of Sum_DP as a group by underlining.

Step 2: Manage equal precedence DPs

Manage the grouped DP_j starting from the second DP in the first underlined group. Check whether exists an $x \in S_DP_j$ of DP_j such that x equals any $y \in S_DP_k$ of its preceding DP (DP_k).

- If yes, put DP_j in the set (i.e. $\{ , \}$) with DP_k .
- If no, move DP_j to a new line under DP_k , and then repeat the following checking process with its current preceding DP (i.e. the preceding DP of DP_k) until it is put into a set or reach the first DP of the group.
- The checking process can be expressed by

$$\begin{cases} \{DP_j, DP_k\}, \exists x \in S_DP_j : x = y \\ DP_k, \forall x \in S_DP_j : x \neq y \\ DP_j \end{cases} \quad (7)$$

where $y \in S_DP_k$ and DP_k is the preceding DP of DP_j .

Step 3: Arrange DPs into U-sets

Arrange all the DPs (DP_p) into sequences starting from the second DP in the first line. Check whether exists a $u \in S_DP_p$ of DP_p such that u equals to $v \in S_DP_q$ of its preceding DP (DP_q) or one of the DPs in the preceding set (DP_q).

- If yes, put a rightward arrow (\rightarrow) between DP_p and its preceding DP or set.
- If no, put DP_p to a new line under DP_q , and then repeat the following checking process until the DP_p is linked by an arrow or reach the first DP in the line.
- The checking process can be expressed by

$$\begin{cases} DP_q \text{ or } \{ \dots, DP_q, \dots \} \rightarrow DP_p, \exists u \in S_DP_p : u = v \\ DP_q \text{ or } \{ \dots, DP_q, \dots \} \\ DP_p, \forall u \in S_DP_p : u \neq v \end{cases} \quad (8)$$

where $y \in S_DP_q$ and DP_q is the preceding DP (or DP in the preceding set) of DP_p .

The flow chart of the steps is shown in Fig. 8.

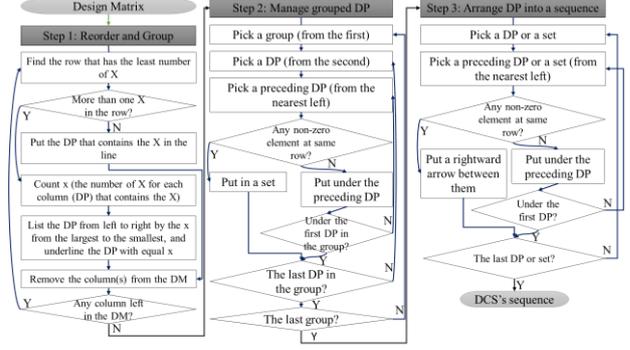


Fig. 8. The flowchart of DCS algorithm.

3.3 DCS Strategies

DCS prescribes some strategies to reduce complexity and increase modularity for three common concept improvement tasks.

3.3.1 For complete improvements of design concepts

After the concept generation process or re-engineering, to refine the concept with the execution sequence, one can follow one of execution sequences to determine DPs. When facing the C-set, one can look up the handbook or searching the existing modules with same functions to see if any similar modules can be retrieved as the best solution (i.e. modular design), or one can eliminate the extra coupling among the DPs in the C-set.

If more than one design are satisfied the FRs in design range, choosing the design with more acceptable execution sequences would result in less imaginary complexity.

3.3.2 For design improvements without DM changes

Sometimes the improvement task is to change a part of the design but the design matrix can remain the same. The scenario could be, for example, to adjust one of the DPs such as out of stock (where the concept is from re-engineering) or difficult to make it real (where the concept is from conceptual design).

For the task, one is suggested to find the changed DP in the U-set, and then follow the sequence in the U to determine the rest DPs. If the corresponding FR of the higher precedence DP in U can't be satisfied due to the change, the complete U-set is suggested to be determined from the first DP in the set.

3.3.3 For design improvements with DM changes

In some cases, the improvement task is resulted in the design matrix change. The scenario could be, for example, to modify the DPs when new FRs involved or some FRs change (where the concept is from re-engineering).

For this task, one is suggested to rebuild the DM with new FRs and new DPs (either expand the design matrix or re-identify the whole DM). Then, one is suggested to follow the strategy of whole design refinement in Section 3.3.1 or the partial design change in Section 3.3.2.

4 Case Study: A Kitchen Faucet Design

Kitchen faucet is a common equipment in a household. As it has been developed for a long time, there are many different designs on the market and are sold as modules. Also, in Suh's Axiomatic Design [4], the faucet problem was used as an example to explain uncoupled and coupled design. In this section, a case of the kitchen faucet design is studied to demonstrate how DCS helps with the functional sets to improve the existing product.

4.1. The Problem Statement

The focused design for the improvement is the kitchen faucet as shown in Fig. 9. By reverse engineering, the design has been transformed into four FRs and four DPs, which are

- FR₁: to control the flow rate,
- FR₂: to control the water temperature,
- FR₃: to control the flow-out direction,
- FR₄: to control the reach-out position,
- and
- DP₁: Cold valve,
- DP₂: Hot valve,
- DP₃: An outlet head,
- DP₄: A hose.

The relationships between FRs and DPs have been identified as shown in Fig. 10. The design ranges for those FRs were assigned as shown in Table 1.



Fig. 9. The original kitchen faucet design. (Kingston Kitchen Faucet, Model # FB2121NDL, \$23.99 at Amazon)

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \end{bmatrix} = \begin{bmatrix} X & X & O & O \\ X & X & O & O \\ O & O & X & X \\ O & O & O & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \end{bmatrix}$$

Fig. 10. The FRs, DPs, and DM of the original kitchen faucet.

Table 1. The design ranges for the kitchen faucet.

FR	Design range
FR ₁	1.0-1.5 ±0.1 gallons per minutes (gpm)
FR ₂	70°F-105°F ±5°F
FR ₃	The tilted angle from -80° to 80° ±5°
FR ₄	In a partial cylinder with a height of 0.30±0.01 m and a 160°±5° sector with radius of 0.15±0.01 m

The design is a partially coupled design because the design matrix is neither triangular nor diagonal. The following section is applying DCS to improve the design.

4.1. The DCS Design Results

The improvement task matches the first improvement task, so the DCS algorithm was directly operated.

In the first step, DP₄ was first picked because FR₄ has the least coupling (only one 'X'), and it is related to DP₄. Then DP₃ was picked because after removing DP₄, DP₃ was the one whose FR has the least coupling. Since the coupled DP numbers of the correlated FRs between DP₁ and DP₂ as well as the number of the correlated FRs between DP₁ and DP₂ are both equal, DP₁ and DP₂ were underlined to note the equal precedence:

$$DP_4 \quad DP_3 \quad \underline{DP_1} \quad \underline{DP_2}$$

In the second step, there checked the grouped DPs, DP₁ and DP₂. Since the coupling was found at FR₁ and FR₂, DP₁ and DP₂ were put in a set:

$$DP_4 \quad DP_3 \quad \{DP_1, DP_2\}$$

In the third step, the checking process was starting from DP₃. One checked the DP₃ is functionally coupled with DP₄ at FR₃, so an arrow was put between them. So far, the line was like:

$$DP_4 \rightarrow DP_3 \quad \{DP_1, DP_2\}$$

Since {DP₁, DP₂} was not found any couplings with DP₃, so they moved to a new line:

$$DP_4 \rightarrow DP_3 \\ \{DP_1, DP_2\}$$

Then, after checking the functional coupling between {DP₁, DP₂} and DP₄, there was no coupling found, so it moved to the place below the DP₄. Since DP₄ is the first DP in the line, the DCS process was complete. The result was shown in Fig. 11.

Basing on the DCS result, one obtained the execution sequences as shown in the equation 9 and 10. The calculated execution sequence number is also two (2! = 2).

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \end{bmatrix} = \begin{bmatrix} X & X & O & O \\ X & X & O & O \\ O & O & X & X \\ O & O & O & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \end{bmatrix} \quad \text{DCS:} \\ DP_4 \rightarrow DP_3 \\ \{DP_1, DP_2\}$$

Fig. 11. The DCS outcome for the studied case.

$$\{DP_1, DP_2\} \rightarrow DP_4 \rightarrow DP_3 \quad (9)$$

$$DP_4 \rightarrow DP_3 \rightarrow \{DP_1, DP_2\} \quad (10)$$

In this case study, the equation 10 was selected as the execution sequence to work on.

With the execution sequence as Eq. 10, DP₄ was improved first to satisfy the FR₄. Among the options DP_{4A} to DP_{4E} (Fig. 12), the DP_{4E} was determined by considering the cost and the design range (Option A and B don't match the design range, and Option E is cheaper than the other two). Based on the determined DP₄, the DP₃ was improved to satisfy the FR₃. Among the options DP_{3A} to DP_{3C} (Fig. 13), the DP_{3A} was determined by considering the DP₄, cost, and the design range (Option

A is the cheapest, and the design range can be satisfied because of the selection of DP_{4E}).



Fig. 12. The DP₄ options from the existing modules. (DP_{4A}: PartsmasterPro Faucet Hose, Model # 58583, \$6.98 at Home Depot; DP_{4B}: Husky Recoil Hose, Model # 4-50E-RET-HOM, \$12.98 at Home Depot; DP_{4C}: Hansgrohe Pull-down Kitchen Faucet Hose, \$35.00 at Home Depot; DP_{4D}: KES Kitchen Faucet, Model # K9250, \$39.99 at Amazon; DP_{4E}: YOSIL Bendable Kitchen Sink Faucets, \$19.99 at Amazon)



Fig. 13. The DP₃ options from the existing modules. (DP_{3A}: Sonline Faucet Aerator, Model # 017097, \$2.18 at Amazon; DP_{3B}: Topbeu 360 Swivel Tap Aerator, Model # LB52, \$3.29 at Amazon; DP_{3C}: WillsCase Faucet Extender, \$8.99 at Amazon)

Then the improving faces a C-set, {DP₁, DP₂}. According to the first strategy of concept improvement task, the strong coupling could be improved by either eliminating the extra coupling terms or considering them as a module to improve together. The following sections demonstrate the improvement with extra coupling elimination via TRIZ method and with existing modules respectively.

4.1.1 Improved Design Results with TRIZ

Theory of inventive problem solving (TRIZ) by Altshuller [20] is widely used for generating innovative solutions and eliminating contradictions in the design. In this case study, the TRIZ approach is used to eliminate the extra coupling. The extra coupling was obtained to be the term between FR₁ and DP₂ after DM rearrangement according to the execution sequence from DCS method (Fig. 14).

$$\begin{bmatrix} FR_4 \\ FR_3 \\ FR_1 \\ FR_2 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 & 0 \\ X & X & 0 & 0 \\ 0 & 0 & X & X \\ 0 & 0 & X & X \end{bmatrix} \begin{bmatrix} DP_4 \\ DP_3 \\ DP_1 \\ DP_2 \end{bmatrix}$$

Fig. 14. The eliminate term in the DM.

To eliminate that non-zero element by using TRIZ, one would have the improving FR₁, which is in terms of number 9, speed, in TRIZ matrix and have the conflict of DP₂, which is in terms of number 17, temperature, in TRIZ. Therefore, the possible principles are 28-Mechanics Substitution, 30- Flexible Shells and Thin

Films, 36- Phase Transitions, and 2- Taking Out. By using the principle 2A of TRIZ, a water heater switch was generated as DP₂*. As a result, the improved DM is shown in Fig. 15.

$$\begin{bmatrix} FR_4 \\ FR_3 \\ FR_1 \\ FR_2 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 & 0 \\ X & X & 0 & 0 \\ 0 & 0 & X & 0 \\ 0 & 0 & X & X \end{bmatrix} \begin{bmatrix} DP_4 \\ DP_3 \\ DP_1 \\ DP_2^* \end{bmatrix}$$

DP₄ → DP₃ → DP¹ → DP₂*

Fig. 15. The decoupled C-set and its execution sequence.

To further improve the design, DP₁ can remain the same as a simple cold valve per the design range, and then the DP₂* is determined as water heater switch (when on, the flow rate is 0.5gpm) in an adjustable temperature to satisfy the FR₁ and FR₂.

Therefore, the final design by the extra coupling elimination is

- FR₁: to control the flow rate,
- FR₂: to control the water temperature,
- FR₃: to control the flow-out direction,
- FR₄: to control the reach-out position,
- and
- DP₁: A cold valve,
- DP₂*: A water heater switch,
- DP_{3A}: A simple faucet aerator,
- DP_{4E}: A bendable hose.

The result is similar to the design in Fig. 16.



Fig. 16. The design result by extra coupling elimination. (A.B Crew Electric Hot Water Heater Kitchen Faucet, Model # COMIN18JU011922, \$49.98 at Amazon)

4.1.2 Improved Design Results by Existing Modules

To improve the U module as well as the C module, {DP₁, DP₂}, one could consider those DPs as an indivisible module to function both “to control the flow rate” and “to control water temperature”. By looking up the existing modules, there are four options (Fig. 17), and the {DP₁, DP₂}_C was selected because of the functional independence and physical integration. Although the design was pointed out that the actual values of flow rate and the temperature were somehow dependent [21], the control operation for the FR₁ and FR₂ are still independent from users’ perspective (because users don’t seriously calculate the temperature and flow rate as long as the temperature and the flow can be varied as the users’ wish during each operation). The final design consequently made the set {DP₁, DP₂} uncoupled (Fig. 18).



Fig. 17. The $\{DP_1, DP_2\}$ options from the existing modules. ($\{DP_1, DP_2\}_A$: DANCO Replacement Lavatory Faucet Handles, Model # 10422, \$10.12 at Home Depot; $\{DP_1, DP_2\}_B$: DANCO Lavatory Handle, Model # 80967, \$9.97 at Home Depot ; $\{DP_1, DP_2\}_C$: American Standard Colony Soft Lavatory Faucet Handle, Model # M961627-0020A, \$14.06 at Home Depot; $\{DP_1, DP_2\}_D$: CHINAURBANLAB Temperature Control Shower Faucet, Model # T21CP32B00-33, no price info.)

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \end{bmatrix} = \begin{bmatrix} X & X & 0 & 0 \\ X & X & 0 & 0 \\ 0 & 0 & X & X \\ 0 & 0 & 0 & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} FR_1, FR_2 \\ FR_3 \\ FR_4 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 & 0 \\ 0 & X & X & 0 \\ 0 & 0 & X & 0 \\ 0 & 0 & 0 & X \end{bmatrix} \begin{bmatrix} \{DP_1, DP_2\} \\ DP_3 \\ DP_4 \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 & 0 \\ 0 & X & 0 & 0 \\ 0 & 0 & X & X \\ 0 & 0 & 0 & X \end{bmatrix} \begin{bmatrix} DP_1^* \\ DP_2^* \\ DP_3 \\ DP_4 \end{bmatrix}$$

Fig. 18. The process of $\{DP_1, DP_2\}$ improvement by existing modules.

The final design by the existing modules is
 FR_1 : to control the flow rate,
 FR_2 : to control the water temperature,
 FR_3 : to control the flow-out direction,
 FR_4 : to control the reach-out position,
and
 $\{DP_1, DP_2\}_C$: An integration module of the up-down flow control valve and left-right temperature control valve,
 DP_{3A} : A simple faucet aerator,
 DP_{4E} : A bendable hose.
The result is similar to the design in Fig. 16.
The final result is similar to the design in Fig. 19.



Fig. 19. The design result by existing modules. (Churun Kitchen Sink Faucet, Model # ZT100, \$32.99 at Amazon)

4.2. Discussion

Both the design results enable the system range to match the design range, and the coupled design has been improved into at least a decoupled design.

In the case of using coupling elimination, the original design that using different proportions of the hot and cold water to achieve the function of “to control water temperature” was changed by using a heater because the hot water valve turned into a switch due to the elimination of the coupling non-zero element. The result shows that the elimination changes the fundamental design. In addition, during the improvement by elimination, the cost of the design couldn’t be considered. As a result, the use of the heater would increase the cost. Therefore, when operating the elimination, the designer is suggested to check the cost and make sure whether the change of the fundamental design is allowed or not.

In the case of using existing modules, the fundamental design wasn’t changed but improved. The function of “to control water temperature” is still satisfied by mixing the hot and cold water in different ratios. That is because the design took the advantages of the existing designs and resources. Taking the C module in DCS to improve the design not thinking the elimination makes the concept improvement of a coupled design in a simpler way with lower cost than the way of eliminating the coupling terms. While the modularity increase, the design was improved with minimal relative complexity.

5 Conclusion and Future Work

The precedence and functional sets were defined to reduce complexity and increase modularity, and the DCS algorithm was modified with functionally independent U-set and insolvably coupled C-set to better obtain the execution sequence and taking the advantage from existing modules. The results of this research have significant impacts on both design theory and design practice. Theoretically, the approach in this research 1) guides designers to improve concepts not only organizing design matrix but also extract additional coupling information to increase modularity and 2) is a more generalized approach than the previous methods that can be applied to any design cases with design matrix. Practically, the research 1) demonstrates the usability of the DCS algorithm and strategies to a kitchen faucet design case and 2) allows the principle of functional dependency and the practice of modular design to be considered simultaneously as much as possible during the conceptual design stage. It is a fundamental contribution that demonstrates how the ideal principles (or axioms) of design theories can be used together strategically with practical design methods (or considerations) in industry practices to generate real-world design results that are both most practical and creative.

For future research, a complete framework for conceptual design would be developed. Also, more design cases would be studied to examine the DCS

method and strategies. Numerical design matrix case would be investigated to see the applicability of the DCS method.

References

1. K. Ulrich and W. Seering, Computation and conceptual design. *Rob. & Comput.-Int. Manuf.*, **4**, 3, pp.309-315 (1988)
2. G. Pahl and W. Beitz, *Engineering design: a systematic approach*. 2nd Ed. Springer (1996)
3. T. McNeill, J.S. Gero, and J. Warren, Understanding conceptual electronic design using protocol analysis. *Res. Eng. Des.*, **10**, 3, pp.129-140 (1998)
4. N. P. Suh, *Axiomatic Design*, Oxford University Press, NY (2001)
5. C.-Y. Wang and S. C.-Y. Lu, The Comparison of Three Different Reasoning Approaches of Dual-Hierarchical Representation in Conceptual Design with Computer-Aided Product and Process Development. In: CIE Grad. Res. Poster Session, 34th ASME-CIE Conference, Buffalo, NY (2014)
6. N.P. Suh, A theory of complexity, periodicity and the design axioms. *Res. Eng. Des.*, **11**, 2, pp.116-132 (1999)
7. D.E. Kim and N.P. Suh. Decoupled Design of Cylinder Liner for IC Engines. SAE Tech. Paper No. 911231 (1991)
8. D.A. Gebala and N.P. Suh. An Application of Axiomatic Design. *Research in Engineering Design*, **3**, 3, pp.149-162 (1992)
9. C.B. Park, D.F. Baldwin, and N.P. Suh. Axiomatic design of a microcellular filament extrusion system. *Res. Eng. Des.*, **8**, 3, pp.166-177 (1996)
10. W. Chen, D. Rosen, J.K. Allen, and F. Mistree. Modularity and the independence of functional requirements in designing complex systems. *Concur. Prod. Des.*, **74**, pp.31-38 (1994)
11. C.-Y. Wang, S. C.-Y. Lu, and A. Liu. Managing Functional Coupling Sequences to Reduce Design Complexity during Concept Improvements. In: ASME 2015 IDETC/CIE, pp. V004T05A034-V004T05A034 (2015)
12. S.C.Y. Lu and A. Liu. A logic-based foundation of axiomatic design. In Proceedings of the 6th ICAD (2011)
13. N.P. Suh. *Axiomatic Design*. Oxford Univ. Press, New York (2001)
14. J.K. Gershenson, G.J. Prasad, and Y. Zhang. Product modularity: definitions and benefits. *J. Eng. Des.*, **14**, 3, pp.295-313 (2003)
15. M.E. Sosa, S.D. Eppinger, and C.M. Rowles. Identifying Modular and Integrative Systems and Their Impact on Design Team Interactions. *J. Mech. Des.*, **125**, 2, pp.240-252 (2003)
16. J.K. Gershenson, G.J. Prasad, and Y. Zhang. Product modularity: measures and design methods. *J. Eng. Des.*, **15**, 1, pp.33-51 (2004)
17. K. Hölttä-Otto, N.A. Chiriac, D. Lysy, and E.S. Suh. Comparative analysis of coupling modularity metrics. *J. Eng. Des.*, **23**, 10-11, pp.790-806 (2012)
18. S. Jung and T.W. Simpson. New modularity indices for modularity assessment and clustering of product architecture. *J. Eng. Des.*, **28**, 1, pp.1-22 (2017)
19. N.P. Suh. *Complexity: Theory and Applications*. Oxford Univ. Press, New York (2005)
20. G. Altshuller and L. Shulyak, TRIZ, 565 the theory of inventive problem solving, Technical Innovation Center, Inc. (1996)
21. J.T. Foley, E. Puik, and D.S. Cochran. The Faucet Reloaded: Improving Axiomatic Design by Example. ICAD 2017, MATEC Web of Conferences. **127**, 01009 (2017)