Assembly and Variety Considerations During Conceptual Design

Saraj Gupta, The Pennsylvania State University
Gül E. Okudan, The Pennsylvania State University
ASSEMBLY AND VARIETY CONSIDERATIONS DURING CONCEPTUAL DESIGN

Saraj Gupta
Pennsylvania State University
University Park, Pennsylvania, USA

Gül E. Okudan*
Pennsylvania State University
University Park, Pennsylvania, USA.

ABSTRACT
Foremost step in the development of any electro-mechanical product is its design, and conceptual design is the most ambiguous and creative phase of design. There exist only a few computational tools that aid designers at conceptual design stage, and mostly designers rely on personal experience or experience of co-workers to generate quality designs. The proposed framework aims at generating robust computerized conceptual designs by incorporating Modularity, Design for Assembly (DFA) and Design for Variety (DFV) principles at the conceptual stage. Conceptual design alternatives obtained from the proposed framework are ranked based on minimum assembly time, and are composed of modules in a way that future changes in customer needs are satisfied only by replacing certain modules. The framework involves searching a design repository of components by using functional-basis and pre-defined graph grammar rules, to generate all possible conceptual design alternatives. These design alternatives are ranked and filtered using a DFA index, and top two alternatives are selected. Selected designs are modularized and filtered using a DFV index to obtain the best design alternative. This paper provides a detailed discussion of the proposed framework, and its working is illustrated through the design of a mounting system for holding a variable message sign, and finally we offer recommendations for future research.

1. INTRODUCTION
Several opportunities exist in the design literature, each individually aiding designers in generating cost efficient designs early in the development of electro-mechanical products. It is estimated that combined effect of all these design approaches will be tremendous, and this research aims at developing a conceptual design tool by integrating DFA, Modularity and DFV. In this paper, we first review the literature related to Modularity, DFA and DFV, followed by a detailed discussion of the proposed framework. The working of the framework is illustrated through the design of a mounting system for holding a variable message sign, and finally we offer recommendations for future research.

2. LITERATURE REVIEW
In design literature, use of function has been recognized as an important part of the design process, with early representations of function found in the field of artificial intelligence [1]. Function can be characterized as a general input/output relation of a system whose main purpose is to perform a task [2]. Researchers at University of Missouri-Rolla (UMR), University of Texas at Austin (UTA), and National Institute of Standards and Technology (NIST) have developed a standard vocabulary of describing the basic functional blocks for design of electro-mechanical products [3]. A tight coupling exists between function and behavior at the conceptual design stage with the overall function being described first, followed by describing the behavior of each component in the context of this function [4]. To enable the generation of conceptual designs, the functional solutions need to be transformed to physical embodiments, and the use of graph grammar rules is widely accepted form for conversion. Graph grammar rules utilize the functional basis [5, 6] of design, and are generally constructed by experienced designers to capture knowledge about a certain type of artifact [7]. The converted physical solutions need to be stored and codified for re-usability at a later stage, necessitating the use of design repositories.

A design repository may be defined as a heterogeneous product design database in which various design solutions can be searched and re-used [8]. These repositories have the capacity to store and retrieve design knowledge and hence, enabling the designer to have easy access to a wide array of design solutions beyond their stored knowledge. During the development of these design repositories, one important consideration is the need to have a standardized function and
component taxonomy [3]. A web based repository has been created by University of Missouri – Rolla (UMR) and in collaboration with the University of Texas at Austin (UTA) and has been refined and populated over the years [9].

Implementation of modular product architecture is observed as one of the responses to challenges faced by mass customization and globalization. Gupta and Okudan [10] conducted a review of existing modularity techniques, and observed that most of the research involving product modularization is carried at detailed stage, with fewer methodologies focusing on conceptual design stage. Manufacturing firms seek to address DFMA to improve the manufacturing situation by changing the product design, thereby increasing productivity significantly without any significant investment [11]. Design for Assembly (DFA) is closely linked to DFMA, and its benefits were not realized until systematic analysis tools became available around 1980 [12]. A quantitative DFA index is proposed by Hsu et al. [13], which is associated with each component that makes up the whole product. The DFA index is calculated based on the ranking module of the product in order to satisfy the future needs. Manufacturing firms would only need to replace the specific parts that are not feasible and modularization is carried at detailed stage, with fewer modularization criteria. As a result, the proposed framework, it is desired to incorporate variety issues along with modularity, resulting in the addition of changes in customer needs as one of the modularization criterion.

Manufacturing firms also seek to optimize the costs of producing all the varieties of products together. Martin and Ishii [15] propose a systematic design methodology leading to a wide coverage of customer preferences, while reducing manufacturing costs, shortening production cycle as well as enhancing product line flexibility. This paved way for Design for Variety (DFV), which seeks to estimate the cost of providing variety at the earlier stages of design. Fujita et al. [16] reviewed product-variety based on multiple considerations like views of customer’s needs, functions, manufacturing modules and hierarchical representation of systems. In the proposed framework, it is desired to incorporate variety issues along with modularity, resulting in the addition of changes in customer needs as one of the modularization criterion. This would enable customer needs likely to change in the future to be clustered together into a module. As a result, the manufacturing firms would only need to replace the specific module of the product in order to satisfy the future needs. Section 3 provides a discussion of the proposed framework.

3. PROPOSED FRAMEWORK

The proposed framework seeks to generate computer aided modularized conceptual designs. The framework is inspired by four independent research methodologies, which are slightly modified in order to integrate them into a robust framework. Figure 1 shows the research tools adapted by the proposed framework.

There are five major steps that need to be executed before the modularized conceptual design can be completely generated. A pre-defined design repository needs to be populated by inputting 1) graph grammar rules, 2) component specifications like image, DFA index, and rule associated with the component and 3) interaction among the various rules. The current framework has been developed using Java Swing within the NetBeans IDE 5.5.1 programming environment. MySQL is used for storing the database tables within the design repository, and JDBC connection is used to link MySQL tables with Java.

The foremost step in the proposed framework is the assessment of customer needs. Fulfillment of all the customer needs may not be technically feasible under the constraints of time and cost. Hence, the customer needs are ranked based on their importance using different tools, and this helps in identifying the important needs that can be fulfilled. Quality Function Deployment (QFD) [17] is an important tool to translate the customers desires into engineering metrics or functions which can then be incorporated into the product architecture. Once the customer needs are obtained, the designers can then develop a black box model, which enables them to list the input / output functions and flows that make up the product. Figure 2 shows a general black box model for an electro-mechanical product.

In the black box model, the overall function to be performed is mentioned inside the boundary of the box, while input energies, materials and signal flows are identified to the left of the box. The output energy, material and signal flows are indicated on to the right of the box, which the remainder flows obtained after the overall function has been executed. The next step involves the generation of the Energy Material Signal...
(EMS) functional model. The EMS functional model is obtained by decomposing the overall function into simpler sub-functions and flows, which are generally described in a verb-object form. These sub-functions and flows are obtained from a standard set vocabulary referred to as functional basis [5, 6]. Figure 3 shows a general EMS model for a general electromechanical product.

Figure 3: A general EMS functional model

The EMS functional model needs to be input into the system in order to generate the conceptual designs by querying the design repository. As a result, a Graphical User Interface (GUI) is developed (adapted from [7]) in which, the functional model is easily input into the computerized framework. While inputting the functional model, each sub-function can be considered as a node, with an input flow and an output flow. These input/output flows can be energy, material or signals, and are obtained from the standard functional-basis vocabulary [5, 6]. Currently, the GUI has the capacity of accommodating up to 50 nodes. The software also allows the designer to save the query and load a previously saved query. After the complete EMS model is input into the framework, the design repository is queried in order to generate all possible conceptual variant designs that satisfy the overall functional model.

In order to automatically generate the conceptual designs, each node of the EMS model needs to be compared with all the nodes of each rule, in order to obtain a direct match. Once the rule gets triggered, all the components associated with that rule are retrieved, resulting in the generation of multiple conceptual designs satisfying the same overall function. In the proposed framework, the design repository comprises of three major database tables: ‘rules’, ‘DFA’ and ‘interaction’.

The ‘rules’ table is used to store the input/output flows and sub-functions corresponding to a user-defined rule. Each rule can have at most three nodes of sub-functions and input/output flows associated with it, and has a unique base id associated with (Figure 4). For the proposed framework, the graph grammar rules are adapted from the research by Kurtoglu et al. [7]. Each rule comprises of up to three nodes of input/output flows and sub-functions. A graph grammar rule may be represented as follows:

R1: IF ((input1 = A1 & sub-function1 = B1 & output1 = C1) & (input2 = A2 & sub-function2 = B2 & output2 = C2) & (input3 = A3 & sub-function3 = B3 & output3 = C3)) THEN RETRIEVE ALL COMPONENTS FROM TABLE DFA WITH baseid = BID;

Here, a number is associated with each rule (e.g., R1). A1, A2, A3 are the three input flows, while C1, C2, C3 are the three output flows and B1-B3 are the three sub-functions of the three nodes. The rules are stored in a catalogued order and are compared with each set of nodes in the query design GUI. Whenever a complete match is found between the antecedent of the graph grammar rule and nodes of the GUI, the rule gets triggered.

The ‘DFA’ table stores the DFA index, image file of the component and also has a unique component ID associated with each component stored in the design repository. For the generation of the conceptual design, each component has to be associated with a rule. Hence, for each component ID, a corresponding base ID has to be selected. Figure 5 shows the GUI for inputting the components in design repository. The DFA index is associated with each component stored within the repository, in order to indicate the ease of assembly of the component with the rest of the product architecture. The DFA index is calculated based on the ranking system developed by Rampersad [14]. In order to calculate the DFA index, values are obtained for 13 different criteria. These criteria include: 1) weight, 2) number of unique components, 3) stiffness, 4) length, 5) presence of base component, 6) vulnerability hardness, 7) shape, 8) size, 9) composing movement, 10) composing direction, 11) symmetry, 12) alignment and 13) joining method. The points are awarded for each of these criteria. For a more detailed explanation of the point criterions, the reader is referred to [18, 19]. The formula for calculating the DFA index is as follows:

\[
\text{DFA index} = 10 \left( \frac{\sum P_i - \sum V_{\text{min},i}}{\sum V_{\text{max},i} - \sum V_{\text{min},i}} \right)
\]

Where:
- \(P_i\) : point value for each criterion, \(i = 1, \ldots, 13\)
- \(V_{\text{min},i}\) : minimum value for each criterion
- \(V_{\text{max},i}\) : maximum value for each criterion

Figure 4: GUI to input the rules into repository

In order to automatically generate the conceptual designs, each node of the EMS model needs to be compared with all the nodes of each rule, in order to obtain a direct match. Once the rule gets triggered, all the components associated with that rule are retrieved, resulting in the generation of multiple conceptual designs satisfying the same overall function. In the proposed framework, the design repository comprises of three major database tables: ‘rules’, ‘DFA’ and ‘interaction’.

The ‘rules’ table is used to store the input/output flows and sub-functions corresponding to a user-defined rule. Each rule can have at most three nodes of sub-functions and input/output flows associated with it, and has a unique base id associated with (Figure 4). For the proposed framework, the graph grammar rules are adapted from the research by Kurtoglu et al. [7]. Each rule comprises of up to three nodes of input/output flows and sub-functions. A graph grammar rule may be represented as follows:

R1: IF ((input1 = A1 & sub-function1 = B1 & output1 = C1) & (input2 = A2 & sub-function2 = B2 & output2 = C2) & (input3 = A3 & sub-function3 = B3 & output3 = C3)) THEN RETRIEVE ALL COMPONENTS FROM TABLE DFA WITH baseid = BID;

Here, a number is associated with each rule (e.g., R1). A1, A2, A3 are the three input flows, while C1, C2, C3 are the three output flows and B1-B3 are the three sub-functions of the three nodes. The rules are stored in a catalogued order and are compared with each set of nodes in the query design GUI. Whenever a complete match is found between the antecedent of the graph grammar rule and nodes of the GUI, the rule gets triggered.

The ‘DFA’ table stores the DFA index, image file of the component and also has a unique component ID associated with each component stored in the design repository. For the generation of the conceptual design, each component has to be associated with a rule. Hence, for each component ID, a corresponding base ID has to be selected. Figure 5 shows the GUI for inputting the components in design repository. The DFA index is associated with each component stored within the repository, in order to indicate the ease of assembly of the component with the rest of the product architecture. The DFA index is calculated based on the ranking system developed by Rampersad [14]. In order to calculate the DFA index, values are obtained for 13 different criteria. These criteria include: 1) weight, 2) number of unique components, 3) stiffness, 4) length, 5) presence of base component, 6) vulnerability hardness, 7) shape, 8) size, 9) composing movement, 10) composing direction, 11) symmetry, 12) alignment and 13) joining method. The points are awarded for each of these criteria. For a more detailed explanation of the point criterions, the reader is referred to [18, 19]. The formula for calculating the DFA index is as follows:

\[
\text{DFA index} = 10 \left( \frac{\sum P_i - \sum V_{\text{min},i}}{\sum V_{\text{max},i} - \sum V_{\text{min},i}} \right)
\]

Where:
- \(P_i\) : point value for each criterion, \(i = 1, \ldots, 13\)
- \(V_{\text{min},i}\) : minimum value for each criterion
- \(V_{\text{max},i}\) : maximum value for each criterion

Figure 4: GUI to input the rules into repository

In order to automatically generate the conceptual designs, each node of the EMS model needs to be compared with all the nodes of each rule, in order to obtain a direct match. Once the rule gets triggered, all the components associated with that rule are retrieved, resulting in the generation of multiple conceptual designs satisfying the same overall function. In the proposed framework, the design repository comprises of three major database tables: ‘rules’, ‘DFA’ and ‘interaction’.

The ‘rules’ table is used to store the input/output flows and sub-functions corresponding to a user-defined rule. Each rule can have at most three nodes of sub-functions and input/output flows associated with it, and has a unique base id associated with (Figure 4). For the proposed framework, the graph grammar rules are adapted from the research by Kurtoglu et al. [7]. Each rule comprises of up to three nodes of input/output flows and sub-functions. A graph grammar rule may be represented as follows:

R1: IF ((input1 = A1 & sub-function1 = B1 & output1 = C1) & (input2 = A2 & sub-function2 = B2 & output2 = C2) & (input3 = A3 & sub-function3 = B3 & output3 = C3)) THEN RETRIEVE ALL COMPONENTS FROM TABLE DFA WITH baseid = BID;

Here, a number is associated with each rule (e.g., R1). A1, A2, A3 are the three input flows, while C1, C2, C3 are the three output flows and B1-B3 are the three sub-functions of the three nodes. The rules are stored in a catalogued order and are compared with each set of nodes in the query design GUI. Whenever a complete match is found between the antecedent of the graph grammar rule and nodes of the GUI, the rule gets triggered.

The ‘DFA’ table stores the DFA index, image file of the component and also has a unique component ID associated with each component stored in the design repository. For the generation of the conceptual design, each component has to be associated with a rule. Hence, for each component ID, a corresponding base ID has to be selected. Figure 5 shows the GUI for inputting the components in design repository. The DFA index is associated with each component stored within the repository, in order to indicate the ease of assembly of the component with the rest of the product architecture. The DFA index is calculated based on the ranking system developed by Rampersad [14]. In order to calculate the DFA index, values are obtained for 13 different criteria. These criteria include: 1) weight, 2) number of unique components, 3) stiffness, 4) length, 5) presence of base component, 6) vulnerability hardness, 7) shape, 8) size, 9) composing movement, 10) composing direction, 11) symmetry, 12) alignment and 13) joining method. The points are awarded for each of these criteria. For a more detailed explanation of the point criterions, the reader is referred to [18, 19]. The formula for calculating the DFA index is as follows:

\[
\text{DFA index} = 10 \left( \frac{\sum P_i - \sum V_{\text{min},i}}{\sum V_{\text{max},i} - \sum V_{\text{min},i}} \right)
\]

Where:
- \(P_i\) : point value for each criterion, \(i = 1, \ldots, 13\)
- \(V_{\text{min},i}\) : minimum value for each criterion
- \(V_{\text{max},i}\) : maximum value for each criterion
The calculated DFA index can have a value from 0 to 10, with zero being the most favorable value for ease of assembly and 10 being the worst value for the ease of assembly.

Figure 5: GUI to input components into the repository

The proposed framework utilizes the rule-based search technique. There may be several cases where more than one component is associated with the same rule. This is particularly useful when the rules are compared with the EMS diagram in order to result in the generation of multiple solutions satisfying the same set of functions. The firing of the rule results in retrieving all the possible component combinations from the design repository that has the base ID equal to the base ID associated with the fired rule. An exhaustive search is conducted, which seeks to combine every component alternative retrieved from a single fired rule to each of the component alternatives retrieved from all the other fired rules, resulting in myriads of combinations.

For example, let us assume that query results in the overall firing of three rules: R1, R2, and R3, where R1 is associated with M1 components from the repository, R2 retrieves M2 components, and R3 retrieves M3 components. Therefore, the total number of combinations generated would be M1*M2*M3. Once the overall component combinations are generated, the total DFA index is calculated for each design by summing the DFA indexes of each component present in that design. The design offering the least total DFA index is ranked 1 and similarly the other combinations are ranked in ascending order based on their total DFA index value. The proposed framework allows the designer to visualize all the generated conceptual design alternatives by viewing their images. The designer then selects two conceptual designs for modularization. In the current framework, it was decided to proceed with two concept alternatives for the sake of simplicity; however, future version may allow the designer to select more than two designs for modularization. It is our hypothesis that a good modularization methodology should cluster components to form modules such that components having higher interactions with each other, and also which are highly likely to change based on future customer needs, are clustered into single modules. The decomposition algorithm [20] (DA) is chosen for the modularization of the conceptual architecture. DA is chosen it is found out to be most suitable modularization technique under the objective metric of “design for assembly” and “design for variety” [21]. DA is a matrix based modularization approach and the two input matrices are interaction and suitability matrices.

The interaction matrix represents the interaction between the components, and it is automatically generated from the interaction table, which is the third table within the design repository. The interaction table stores the interaction amongst the different rules stored within the repository. The designer needs to determine whether there is any interaction between the rules and also needs to indicate the direction of interaction. For example, if there is an interaction between rule1 and rule2, it is denoted by 1, otherwise 0. The interaction table is useful for modularization of the conceptual designs based on Huang and Kusiak’s decomposition algorithm [20]. The interaction matrix is developed while the designer examines the dissected components during population of the repository. These interactions can be either flow of electricity between the components, or the force exerted by one component towards the other. The direction of the interaction is important as some interactions are unidirectional, while others are bi-directional. Accordingly, they are implemented in the decomposition algorithm. Figure 6 shows the GUI of the interaction table.

Figure 6: GUI to input interaction matrix into repository
On the other hand, suitability matrix represents the suitability for components for inclusion in a module. The suitability matrix is generated at runtime since it may be possible that two components interact with each other, but they may not be suitable for inclusion within a module. The suitability matrix is also dependent on the customer needs to a great extent. Suitability matrix is generated by asking the designer whether the components within each of the two selected designs, are suitable to be included within the same module. Unlike the interaction matrix, direction is not important for the suitability database. The suitability level for component1 to be included along with component2 is mentioned in column 3, which suitability may be left blank, if it does not make any difference if the two components are included within the same module. Figure 7 shows the different levels of input for the suitability matrix [20].

<table>
<thead>
<tr>
<th>Input letter</th>
<th>Corresponding Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>strongly desired</td>
</tr>
<tr>
<td>e</td>
<td>desired</td>
</tr>
<tr>
<td>o</td>
<td>strongly undesired</td>
</tr>
<tr>
<td>u</td>
<td>undesired</td>
</tr>
</tbody>
</table>

**Figure 7**: (a) Input levels for suitability matrix (b) GUI

Once the suitability and interaction matrices are defined, the decomposition approach is then followed in order to transform the interaction and suitability matrix, allowing one to explore the potential modules amongst components. Triangularization of the interaction matrix is done based on with the algorithm present in Kusiak et al. [22] and the suitability matrix is rearranged so that the sequence of rows and columns in both the matrices remains the same.

The triangularized interaction matrix and rearranged suitability matrix are then combined to form the modularity matrix. Components are removed from the module if they are either undesired for inclusion in the module or interact with the remaining components in the module to a degree lesser than the first component. These removed components are placed at the end of the modularity matrix and this process is repeated until no more components can be deleted. On the other hand those components which are used and strongly desired for simultaneous inclusion in two modules are duplicated and this process continues until no more components can be duplicated. The two user selected modularized conceptual variants are then analyzed and secondary filtering is done to select the best modularized conceptual design. This filtering criterion is known as DFV index (i.e., Generational Variety Index (GVI) adapted from [23]). It is an indicator of the amount of redesign required for each component within a product in order to meet the future market requirements. Hence, the DFV index is basically an estimate of the required changes in a component due to external or non-controllable factors [23].

The two design alternatives are evaluated and the design which generates the least DFV index is selected as the best design. In order to quantify these customer needs, required for generating the DFV index, a two phase QFD technique is adapted from the research by Martin and Ishii [12]. In the first QFD phase, a relationship is developed between the customer needs and the engineering metrics (EM) [23]. The engineering metrics are measurable items, which are translations of subjective customer needs into quantifiable engineering specifications. Figure 8 shows a general QFD-I matrix. The next step involves estimating the qualitative estimation (High/Medium/Low) of the range of change of customer needs [23]. This step enables the design team to begin thinking about how the customer needs change with time. It is desired that the

**Figure 8**: A General QFD Phase-I matrix
customer needs which are forecasted to change significantly with time, should be restricted to only one module and not many modules. Otherwise, all modules need to be re-designed in order to accommodate the future customer needs. If the range of change of customer needs is high, then it is denoted by 3. If the range of change is medium, then it is denoted as 2, while low range of change of customer needs is denoted 1.

The next step is to generate the QFD Phase II matrix, which maps the engineering metrics from phase I to the modules used in the design. A general QFD Phase II matrix is shown in Figure 9. The final step involves the development of the DFV matrix and calculation of the DFV index. The total DFV index value is obtained by summing the last row of the DFV matrix for each concept variant. The conceptual design having lower DFV index value is best overall concept, since lesser redesign effort would be required within its architecture in order to meet the future customer requirements. Figure 10 shows a general DFV matrix with the index value associated. In Section 4, the working of the proposed framework is illustrated through the design of a mounting system for holding a variable message sign.

### Figure 11: Hierarchical customer needs

Next, a black-box model of the VMS mount system was prepared. The main objective of the mounting system is to secure the VMS, and this objective is reflected in the black-box model (Figure 12). The black box model is then used to create the EMS diagram, which is obtained by decomposing the overall function into simpler sub-functions and flows. This EMS diagram of the VMS mounting system is shown in Figure 13, which needs to be input into the EMS GUI in order to generate the conceptual designs by querying the design repository.

### Figure 12: Black box for the VMS mount system
Analysis of EMS diagram reveals four different steps associated with the mounting system design. First step governs the use of Human Energy (HE) to fixture the base mount of the mounting system, either on the car or on ground. The second step involves utilization of HE for height adjustment of the base mount system, so that VMS can be easily noticed by the passing commuter vehicles. Once height of base mount is adjusted, VMS fixture can be assembled with the base mount. Lastly, the VMS is coupled with the VMS fixture so that it is secured thoroughly on to the ground/car. Corresponding to these, four graph grammar rules are generated and are shown Figure 14.

The design team begins to brainstorm all the components that satisfy the different sub-functions and flows shown in the EMS diagram. These components will be stored in the design repository for later reuse. After identifying the components, a unique ID is associated with each of them and their DFA indexes are calculated based on the 13-point ranking system. Figure 15 shows the identified components for the mounting system along with their picture, associated rule, unique ID and DFA index value. The design team also analyzes the interactions between the four rules in order to populate the ‘interactions’ table. Directionality of the interaction is also important and if there is an interaction from one rule to another, it is denoted by * as shown in Figure 16. The design repository is now assumed to be populated with rules, interactions and component specifications.

The next step involves the comparison of the EMS model with the rules stored in design repository. In the design repository 9 components are associated with ‘holding the sign’ rule, 5 components with ‘attaching the sign to the base frame’ rule, 7 components with ‘extending or collapsing’ rule, and 8 components with ‘mount to ground/vehicle’ rule. In total, this
allocates for 2520 conceptual design combinations satisfying overall function of “securing VMS”. These combinations are ranked based on their total DFA index, and the design team selects two conceptual designs (Figure 17). In both the selected designs, it is assumed that ‘Mou_ang_2’ can be mounted on the ground as well as on the rear trunk of the stationary vehicle. The sign can be mounted on rear trunk of the vehicle through a pair of suction cups, which are fixed on the base of “Fra_rot_1”.

Figure 17: Conceptual designs selected after primary filtering

The design team then inputs the suitability matrix for both the selected concepts (Figure 18), and modularization of both the selected designs is carried out. Implementation of DA resulted in the generation of two modules for concept #1 and four modules for concept #2. These are shown in Figure 19.

Figure 18: Suitability matrices for both selected designs

In order to obtain the best design, secondary filtering is done using the DFV index. Figure 20 shows the QFD phase I matrix while 21 shows the QFD phase-II matrices for the two concepts.

Next step involves the calculation of the DFV index (Figure 22). DFV index for concept #1 is found to be 16, while for concept #2, DFV index value is 32. Hence, concept #1 (Figure 23) is chosen as the best overall concept, since less redesign effort would be required to meet the future customer needs.

Figure 19: Modules obtained for the two selected designs

Figure 20: QFD Phase I matrix

Figure 21: QFD Phase II matrices for the two selected concepts

Figure 22: DFV matrices for the two selected concepts

Figure 23: Final sketch of the best selected design concept

5. CONCLUSIONS AND FUTURE SCOPE

In this research paper, a framework has been proposed for the generation of modularized conceptual designs by considering DFA and DFV issues. The working of the framework has been illustrated through the design of a mounting system for holding a variable message sign. Future improvements to the proposed framework include the
replacement of the component image files with actual 3D CAD models. This feature would enable the designers to utilize the 3-D features of CAD at the conceptual design stage itself, thereby eliminating thereby eliminating the shortcomings of existing CAD technology [24]. Effectiveness of the proposed framework will also be evaluated by comparing the conceptual designs generated from framework with the conventional paper-pencil designs generated by designers.

REFERENCES