

DETC2008-50005

CUSTOMIZING PRODUCTS USING FUNCTIONAL COMPONENT MATRICES

Ali Kamyab

Graduate Research Assistant
University at Buffalo - SUNY

Department of Mechanical and Aerospace Engineering
Kamyab.ali@gmail.com

Kemper E. Lewis

Professor

University at Buffalo - SUNY
Department of Mechanical and Aerospace Engineering
Corresponding Author: kelewis@buffalo.edu
Tel: (716) 645-2685

ABSTRACT

Modern design methodologies have used Function Component Matrices in a variety of different ways in order to support various facets of an engineering design process. The mapping of functions to components can be used to model and capture the dependencies and relationships that exist. This process is accomplished by breaking down complicated functions into smaller, easier to understand functions. This decomposition allows engineers to get a better understanding for how a change in each component within a product will affect the overall operation of the product. Being able to recognize the impact of the propagation of a sub-function change will give designers a better understanding of the flexibility (or lack thereof) of choices they have when designing a product for customization. In turn they can be used to inform the consumer regarding the consequences their customization choices can have on the final product. This paper discusses how a Functional Component Matrix (FCM) can be used to assist in this process of product customization and understanding change propagation.

1. INTRODUCTION

Preferences tend to vary when two individuals look at a product. Generally speaking, no two people have the same preference structure. As corporations have grown in size so has their sight on the fact that every customer is unique [1]. Most products are designed to satisfy a wide range of preferences, and thus are not designed specifically for one individual. When a product is not designed specifically for a certain customer, there is a certain degree of compromise with respect to meeting each customer's preferences precisely.

While tradeoffs will exist when designing any product, if these tradeoffs can be minimized it is likely that the product will be more successful. In fact, in [2] it is implied that some performance tradeoffs are a result of poorly designed systems.

The number of necessary tradeoffs can be reduced by allowing the consumer to make some of their own product configuration decisions, thus creating customized products and allowing them to make their own individual tradeoffs and in the process designing better fit systems.

Allowing a consumer to create their own product configuration would require the involvement of the consumer in portions of the design process. Customers have traditionally been brought in early in a design process for market testing and focus groups [3], or during the later stages of design for testing purposes [4], but have had limited involvement with the embodiment or detailed design phases. In addition, in [5], it is noted that the customers who provide inputs to product design are not necessarily the product purchasers, implying that design is not truly customized.

Going from conceptual design to a final product typically requires the extensive involvement of several specialized individuals (i.e. engineers of various disciplinary fields), but some previous work has attempted to involve the customer more in a product design process. "It is clear that future organizations will seek to achieve far greater levels of customer involvement, culminating in continuous customer engagement at all stages of manufacturing" [5].

Recognizing the challenges to having customers involved throughout a design process, a company can instead take a product they wish to design and use possible conceptual design choices as customer options, instead of picking one design as a solution. By doing this they leave the final design option to the consumer and are able to better address the preference needs of the consumer. The consumer can then choose between these conceptual options (that have realizable embodiment) instead of being forced to pick the design option chosen by the engineers. This idea, however, needs formality and requires a format in

Class	Material	Signal		Energy	
Secondary	Human	Status	Human	Electrical	Mechanical
	Gas	Control	Acoustic	Electromagnetic	Pneumatic
	Liquid		Biological	Hydraulic	Radioactive
	Solid		Chemical	Magnetic	Thermal
	Plasma				
	Mixture				

Table 1 - Flow Class Functional Basis [15]

Class	Branch	Channel	Connect	Control Magnitude	Convert	Provision	Signal	Support
Secondary	Separate Distribute	Import Export Transfer Guide	Couple Mix	Actuate Regulate Change Stop	Convert	Store Supply	Sense Indicate Process	Stabilize Secure Position

Table 2 - Function Class Functional Basis [15]

which it will be conducted. In this paper, we attempt to provide the first steps to providing this configurable capability by, introducing a collaborative design process that uses FCM's to help engineers create customizable products, while giving consumer great control over a products sub-functions.

In Section 2 some background on functional modeling and functional component matrices is presented building the groundwork that our method was based on, followed by a review on the current state of customization within products. Section 3 formalizes how functional component matrices can be used to facilitate a customization interface, describing some technical issues that existed and how they were resolved. In Section 4 a case study is presented to demonstrate the method, and finally the paper is concluded with the impact of the study.

2. BACKGROUND

2.1 Functional Modeling

"In engineering design, the end goal is the creation of an artifact, product, system, or process that performs a function or functions to fulfill customer needs" [6]. Chenhall takes it as far as saying the lexicon which is used to classify items in a museum "is based on the assumption that every man-made object was originally created to fulfill some function or purpose and, further, that original function is the only common denominator that is present in all of the artifacts of man, however simple or complex" [7]. While some simple products may have just one function for the consumer (i.e., a house key used for locking and unlocking a door), other more complex products with challenging functions may require many "sub-functions" [8] to complete its main function. These sub-functions are an integral part of engineering design today. Most conceptual design methods that are used rely on some form of functional decomposition to carry out the method [4,-9-11]. However, in order to decompose a product's overall

function into meaningful sub-functions, a formal method is required.

The need for formality of this process has pushed research into developing a high level design language to describe product functions, allowing a systematic approach to functional modeling. The result of some of this work has created a design language known as the Functional Basis [6]. The Functional Basis was developed by combining two similar, independent research efforts [6, 12]. Other approaches exist including the work in [13] that uses the 20 subsystem representations from living systems theory to represent mechanical design functions. Kirschman and Fadel [14] have proposed four basic mechanical functions groups. Their approach however tends to vary from the standard verb object sub-function description common to most other methodologies.

The standard Functional Basis consists of a flow class (Table 1), and a function class (Table 2) that use flow and function words to describe functions and sub-function within a product. All together there are three flow classes: *material*, *signal*, and *energy*. The eight function classes are *branch*, *channel*, *connect*, *control magnitude*, *convert*, *provision*, *signal*, and *support*. These classes and flows are believed to be broad enough where they cover all aspects of engineering design, and can be used to describe any product sub-function in a generic form.

There are three tiers of sub-functional breakdown [6], and depending on the depth in which a designer wants to decompose a product's overall function, more layers can be used. A single level breakdown can be used for simple products, like electronics (e.g., digital camera or cell phone), and for larger products (e.g., automobile engine) a tertiary level may be necessary. The three levels that exist are the class level, the first level, which represents the overall function of the product. The subsequent secondary and tertiary levels

represent the overall sub-functions broken down in greater detail. Throughout this paper, and in our case study, we maintain a secondary tier functional breakdown. The second tier breakdown gives sufficient sub-function information to be able to implement the methodology.

Once the sub-functions in a product are found, they need to be presented in a useful manner in order to use them effectively. One way in which function decomposition has been used is in Functional Component Matrices.

2.2 Function Component Matrix

As stated, a complex overall function can be broken up into sub-functions. Pahl and Beitz describe the process of breaking down complex functions as having these two parts.

- The determination of sub-functions facilitating the subsequent search for solution; and
- The combination of these sub-functions into simple and unambiguous function structure [9].

When describing a product and the function of that product, the sub-functions within a product can have many levels [16]. Figure 1 shows the overall function and sub-function hierarchy.

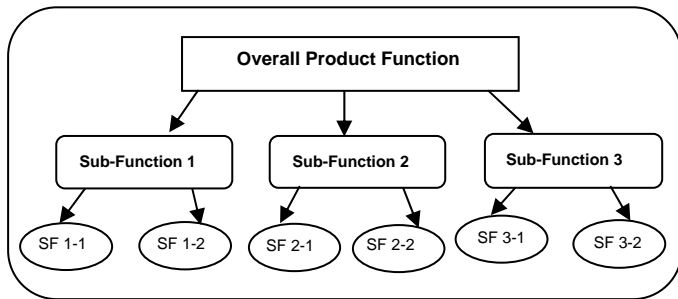


Figure 1 - Functional Decomposition

To begin the creation of an FCM we require all the sub-functions of a product, along with all the components that are used to fulfill it. The purpose of a FCM is to represent components which solve given functions where the $(i, j)^{th}$ entry in an FCM represents whether function i is solved by component j [17]. Using the Functional Basis discussed in Section 2.1 a product's overall function can be decomposed into sub-functions. Having generated all sub-functions for a product, the next step in creating a FCM is to come up with the components that will fulfill the sub-functions.

Using the sub-functions for a given product and all the corresponding components, an FCM can be developed. Once the matrix has been populated with all the sub-functions and components the relationships between them needs to be addressed. This process requires checking to see which component will solve which sub-function. Each component sub-function relationship is assigned a value of one if the

component solves the sub-function and a value of zero if it does not. This mapping of sub-functions to components makes the relationship between them more obvious.

Sub-functions	Components			
	Component 1	Component 2	...	Component n
Sub-function 1	1	0	0	0
Sub-function 2	0	1	0	1
Sub-function 3	1	0	0	1
...	0	0	0	0
Sub-function m	1	1	0	1

Table 3 - Example FCM

Table 3 shows a simple example of an FCM. The method starts by having all the sub-functions of the product in the first column of the matrix. The remaining columns contain the product components. The entries in the matrix show whether a relationship exists between the sub-function and the component. An entry of one lets the user know that a relationship exists, while an entry of zero states that there is no relationship.

From the first row it is evident that sub-function 1 is only coupled to component 1. This is apparent by the only entry in the first row. This entry is in the second column and has a value of 1. The other entries for component 2 and n are both zero, meaning they have no effect on sub-function 1. Sub-function m is coupled to all the components in the product. Any change in one of the three components in this product will have some effect on that sub-function. This type of binary relationship in FCM's has been used in different areas of engineering [18].

FCM's have served engineers in a variety of different ways [16, 19-21], and can be a powerful tool if used appropriately. Different approaches have been used for the rating scheme in a FCM depending on what it's being used for. For example, in [22, 23], the functional representation of a product stores customer needs and functions in the matrix instead of just the relationships between components and functions. A tri-level system was used to describe the sub-function component relationship in the method purposed. This paper demonstrates how FCM's can be used to help customize a product, but first the current state of customization within products is discussed.

2.3 Current Customization Methods

Currently many companies offer customization in a variety of different ways. Pine and Gilmore [1] discuss four approaches to customization and how they are used today:

- **Transparent** – Where the customer does not know that the company is researching their preferences, and constantly making changes to the services and products they offers.

- **Adaptive** – Here customization is achieved through some form of modification, or reconfiguration that is done to the product after it has been produced.
- **Cosmetic** – The simplest form of customization where customers make choices on how the final product will look.
- **Collaborative** – This type of customization permits the customer to have some participation in the design process, thus creating a product based on their individual preferences.

Using the method described aids in the design of reconfigurable systems.



Figure 2 - Examples of Expand/Collapse [25]

Transparent customization fulfills the needs of an individual customer without the customer knowing that the product has been customized for them. Instead of investing the time with a customer to describe and address their preferences, transparent customizers observe the behaviors and routines of their customers and try to predict their preferences over a period of time. This type of customization works best when the customers' specific needs can easily be predicted. Transparent customization also can work when the customer is not interested in explicitly stating their own preferences and may not want to be bothered with surveys on preference structures. One example is a company called ChemStation which offers soap to commercial establishments like car washes [1]. ChemStation has independently analyzed each customer's needs and knows precisely when they run out of soap by analyzing their usage patterns. This gives them the ability to deliver soap to the customer when they need it without the customer ever ordering more.

A second approach to customization is an *adaptive* approach. Rather than provide a customized product, adaptive customizers tailor their product so that it can easily be modified, or reconfigured to fit different functions without any direct interaction with the company. In this approach the company releases a standard product, but once the product is released it can be altered. This approach works best if the prospect customers wish to have a product that can perform different tasks based on different situations. An effective realization of this type of customization is reconfigurable systems, which are systems that are designed to maintain a high level of performance through adaptations in their configurations when operating conditions change [24].

Designers have taken ideas from nature amongst other things to create products that change functionality by reconfiguring themselves. The puffer fish, shown in Figure 2 adapts by expanding if it feels threatened. The fundamental transformational principle behind this adaptation – expand/collapse [25] – has also been used in consumer products, as illustrated by the portable sports chair and the collapsible hand bag/towel shown in Figure 2. In [25], a group of comprehensive transformation principles and facilitators are developed to support the design of a reconfigurable system.

A third approach is *cosmetic* customization. In this approach the customer is given a standard product that only differs in the way it looks. This type of customization should be adopted by a company if their product satisfies almost every customer and only the product's form needs to be customized based on preferences. All functions performed by the product remain the same. This approach is used when customer do not require different functionality from the product however wish for their product to look different than all the others. As an example of cosmetic customization Puma has available on their website its Mongolian shoe barbecue [26], giving customers the ability to pick from several different colors to place anywhere on a pre-selected shoe. Other companies like Nike and Reebok now offer the same type of cosmetic customization on a limited number of shoes. Cosmetic customization can typically be accomplished using simple aesthetic changes that do not impact functionality in any way. This is the main reason that most cosmetic customization will come towards the “end of the value chain” [1].

Hardly any company offers a *collaborative* customization process (a fourth approach) where specific performance issues can be changed for a product. Certain companies have taken this idea forward and now offer made to order products (e.g. jeans, bicycles) that have physical dimensions that cater to customer preferences. However the product architecture is still fixed [5].

Other companies offer functional customization in the form of product platforming, product families, and modular options [27-29], which is more of an adaptive approach. While improving economics and scope in a manufacturing process a product platform also can facilitate customization by allowing products to quickly change to satisfy different needs for consumers [30]. Product platforms serve many purposes and offer great variety when it comes to products, however offering variety in a product is not the same as offering a customizable product. Variety of products will give customers choices, but too many choices can have negative effects [31, 32, 33], and does not allow a consumer to implement their own preferences structure on a product.

To achieve functional customization, these rigid platforms would have to be made more and more flexible essentially to the point of having no broad platforms, but customizable modules and component architectures. This would then allow the consumers to pick any option on a product, and the options would not be limited based to a certain product level. To facilitate this level of customization, the designer must have full understanding of how each component within a product impacts the products functionality. Furthermore, the designer must know what type of change propagation occurs in a product with each customization choice. The focus of this study is to help establish some foundational methods for configurable design. In the next section we propose a method to carry out a collaborative customization process.

3. COLLABORATIVE CUSTOMIZATION METHOD

3.1 Product Decomposition

3.1.1 Step 1 Develop functional basis of product

Using the Functional Basis technique presented in Section 2.1, a functional decomposition of the product is created. This gives a list of all the sub-functions that exist in the product, which are used in the FCM matrix. The depth of the decomposition depends on the complexity of the product and the level of customization desired for the product. This can generally be done during the conceptual or embodiment design phases if the product is in development. If an existing product is being used then the decomposition can be performed using the product itself. Within this phase the designers can start considering what sub-functions in the product are viable options for customization. At the completion of this step a list of sub-functions is generated. The next step is to create a list of components that will carry out the sub-functions.

3.1.2 Step 2 Develop component list

If working with an existing product, a component list most likely exists for the product, however if for any reason one is required a physical dissection of the product can be used to construct the component list. Conversely, if the product is being developed, a conceptual functional analysis can be utilized to map all the sub-functions to a set of solutions [4]. Then, the solution set can be mapped to a list of components. In both cases this step results in a list of components for the product's sub-functions. The sub-functions and components are input into the FCM and the relationships between them are determined.

3.2 Using FCM for Customization

3.2.1 Step 3 Populate FCM

Having developed a set of sub-functions and components the FCM can now be assembled. All the components and sub-functions are translated into matrix form. Now the designer is

tasked with defining the relationships that will describe the connectivity between sub-functions and components. The definition of relationships is essentially the process of labeling what component will solve what sub-function. Multiple methods have been used for the rating scheme, and for this study we have developed a new method, catered to product customization. In Section 3.5, the new rating scheme is introduced.

3.2.2 Step 4 Using FCM Relationships to Develop Customization Questions for Use in the Interface

Once the FCM is populated the engineer can look for possible customizable sub-functions. Knowing the dependency between the sub-functions and components can help facilitate the creation of a product that can be functionally customizable. In order to create a customizable interface, a series of questions pertaining to the sub-functions needs to be created. While inspecting the sub-function component relationship the engineer can create a series of customizable questions for use in the interface. These questions are used in the interface through a series of line commands that the consumer uses to make changes on the product. The questions focus on the sub-functions in the product and not the components. They can also change the level of that sub-function (i.e. battery life, storage) where a choice of more or less is an option.

It is within these sub-functions, and the ability to change the sub-functions that a designer can give the consumer the ability to functionally customize their product and create a collaborative customized product. The questions should translate the sub-functions into customer attributes and then leverage the FCM to determine the necessary component changes, as explained in the next section. The interface also includes a feasibility check which is used to check for any preference inconsistencies that the consumer may have stated while customizing the product. This check is demonstrated in Section 4.3.

3.3.2 Step 5 Detect dependencies and update product

With the FCM in place it is insured that any customer change will be functionally feasible. This process is done by creating an interface that is checking the product's FCM after every customization choice made by the customer. As the customer changes the sub-functions (e.g., add or take away sub-functions) in the product the interface checks the FCM and the sub-function component dependencies to make sure that the product is still functional. When a sub-function is removed by the customer, so is the component that is responsible for solving it. When this happens the FCM is checked for other dependencies the component that was taken out may have with other sub-functions and make the consumer aware of the consequences of their actions. When a sub-function is added the dependencies in the FCM will get updated to reflect this choice. The collaborative design process ends at this stage

however a modified FCM is required to carry out this process which is presented in the next section.

3.3 Modified FCM for Customization

For a customization interface, it is necessary to know the strength of dependence rather than just the existence of one. The strength of dependency is required to give the designer and the consumer a sense for how much a change in a sub-function would affect a product. Using a binary rating would only capture that a sub-function was affected, but would not capture how much it would be affected. Using a tri-level method gives more qualitative information which serves the purpose of this study better.

This method describes the dependency of each component to each sub-function in three tiers. The levels 0, 1, 2, relate to how much each component impacts each function. The levels essentially pertain to how much the product is affected by a customization choice, and lets the user and the designer know if the product will be able to function based on the choices they have made. The rating scheme was chosen to show an increase in the dependency level. Only 3 levels of dependency are needed for the method. The only concerns that will have to be reflected in the FCM can be described in three levels. A more detailed rating scheme having more levels could cause confusion for the consumer and the engineer. The three levels are:

- **Level 0** – No relationship exists between the component and the sub-function.
- **Level 1** – A relationship exists between the component and the sub-function, however the component is not the main component that fulfills the sub-function.
- **Level 2** – The component is the primary component fulfilling the sub-function.

A digital camera can be used to demonstrate the tri-level relationship. Table 4 illustrates a simplified FCM which has a sample of digital camera sub-functions and components. The table shows that of the given components, the lens is the most vital to sub-function 3 (Zoom), indicating that zooming is critically dependent upon the camera's lens.

Sub-functions	Components		
	LCD Screen	Lens	Memory Slot
Capture Video	1	1	2
Picture Editing	2	0	0
Zoom	0	2	0

Table 4 - First Three Rows Updated

A customer can customize the sub-function on the camera by stating they do not wish to have Zoom as an option with their camera. This change would have the largest impact on

component 2 as seen in Table 4. When the customer states zooming is not required, the FCM predicts that component 2 would not be fulfilling any function and therefore could be taken out of the camera. With this customization, the camera can still take pictures and perform all the other sub-functions. Note that this component may be added later due to other dependencies or preferences. If a component is related to more than one sub-function and if any of these other sub-functions are added during the customization process, then the component will be added as well. The product would then have all the functionality providing by the component even if the customer has not stated a preference for some of the functions.

This raises another issue. Component 2 has a level 1 dependency to sub-function 1. In a more complicated example it may have other level 2 dependences and thus make other sub-functions nonfunctional. Level 0 dependencies are not a concern, as they do not affect the functionality of the product. Here is where the FCM becomes very powerful in the customization process. After every customer customization choice the algorithm will check all the relationships in the FCM to determine what components have been affected and provides the consequences of each choice on the final product to the user.

As an example in Table 4, once component 2 is removed from the product the algorithm searches other sub-functions to determine what components are affected. If any other level 2 dependencies exist in the product for that specific component, the algorithm will determine that those sub-functions are no longer available. The consumer is also made aware of level 1 dependencies. This is done by stating that some sub-functions have been limited by this choice, but not completely removed from the product.

In this case we see that sub-function 1 (Capturing Video) has a level 1 dependency on component 2. The algorithm now informs the customer that the ability to capture video has been limited. This is with the assumption that Capturing Video could have used zooming while taking video and does not require zoom to actually take any video. This is a limiting relationship and the lens is not the main component in the camera responsible for capturing video. This limitation on a sub-function is what a level 1 relationship would describe. By deleting this component a sub-function was not taken away from the product, but was limited in some way.

If the user decides to have the sub-function Zoom as part of the product, the interface will take all the dependencies for this sub-function and the related components, and make the dependencies a level 0. All the entries in the sub-function Zoom column will be changed to a level one dependency. This is done to help assist the customization process. By changing the dependencies in the FCM for these particular areas of

concern, later customization choices will not be affected by the old dependency relationships, and are now only affected by the new developed dependencies based on the customer's preference. Using FCM's and the tri-level rating scheme the designer and the customer can be assured that the final product of choice will be functional.

4. CASE STUDY

4.1 Developing a FCM for an iPod

The idea of having a product that is unique to the individual has the benefit of fitting that person's performance criteria, as well as having a product that is completely unique to them. In order to implement this methodology an Apple iPod™, shown in Figure 3, is used for the case study. The product was chosen for its popularity as well as the multiple models that are available for the product at this point. This makes the product ideal for a customization case study.



Figure 3 - Standard Apple iPod™

The first required step in creating the customization interface is to develop an FCM for a standard iPod. The Functional Basis along with the functional modeling discussed in Section 2.1 are used to decompose the product into a second level sub-function decomposition. Maintaining a second level sub-function breakdown was sufficient as these sub-functions would allow the customer to have full control over the iPod's functionality. It is determined that eight sub-functions exist in a standard iPod and it is these sub-functions that are used to construct the FCM:

- Play back music
- Play back video
- Store data
- Receive energy
- Store energy
- Display Information
- Receive user input
- Sync

The next step in constructing the FCM is to generate a list of components within the iPod that solve the sub-functions. With a product that currently exists reverse engineering can be used to develop the component list. In a standard iPod, 11 components are found. A complete FCM using the tri-level system is shown in Table 5. This table is the initial FCM for the iPod and the starting point of customization for the product. As the consumer makes customization choices the FCM is updated. As shown in Section 3, as sub-functional level choices are made the relationships are checked to see if any conflicts exist with the choices.

If the user decides not to add functionality the FCM remains unchanged and the algorithm will just do a simple check to see if there is any level 1 or 2 conflicts in any of the columns relating to that sub-function. If the user adds sub-function level functionality to the product then the relationships in the FCM will need to be updated to reflect these preferences.

4.2 Implementing iPod FCM in the Customization Interface

The FCM was loaded into a program (coded using C++) in matrix form. Once the program is loaded the customer is given a series of questions using command lines, relating to the different choices they have on the iPod. These questions were geared towards the iPod case study but some of them can be generalized to fit other situations. The questions are as follows:

1. *Will you be watching video on your iPod?*
2. *What type of user interface do you prefer?*
3. *How much storage space do you require?*
4. *Is the size of the iPod an issue and if so do you prefer a smaller or larger iPod?*
5. *Do you wish for your iPod to be portable?*

The questions are presented in line commands within the interface and the user picks between predetermined answers. Their answers are not related to any specific component. However, the questions are mapped to the functions using the FCM. The program then executes and gives the user an initial iPod based on the preferences they have declared.

The questions are chosen with the intent of covering all the sub-functions in the FCM to a customized iPod based on their preferences. For example the first question, "*Will you be watching video on your iPod*", relates to four of the eight functions in the FCM. The functions are; playback music, play back video, display information, and receive user input. Similarly other questions map to other sub-functions and each sub-function is referenced at least once when all the questions have been answered.

Functions		Components										
		c1 LCD Screen	c2 Video decoder	c3 Audio decoder	c4 Power manager	c5 audio jack	c6 hold button	c7 USB connection	c8 Storage device	c9 case	c10 wheel	c11 touch screen
F1	play back music	1	0	2	2	2	0	0	2	1	1	1
F2	play back video	2	2	2	2	2	0	0	0	2	0	0
F3	store data	0	0	0	0	0	0	2	2	1	0	0
F4	receive energy	0	0	0	2	0	0	2	0	1	0	0
F5	store energy	0	0	0	2	0	0	2	0	1	0	0
F6	display information	2	2	0	2	0	0	0	2	2	0	1
F7	receive user input	1	1	0	2	0	0	0	0	1	1	1
F8	sync	0	0	0	0	0	0	2	2	1	0	0

Table 5 - iPod FCM

The customer makes customization choices by answering the line command questions and in turn, changing the sub-function level functionality of the product. The FCM is updated and the product is checked for functionality after every choice. Once the customer has made all the desired choices a final check is done by the program to see if the iPod will function. This is done using a feasibility check, which tests the iPod for consistency. The main test is checking the overall product function and making sure the customer has not stated any inconsistent preferences.

Not all the sub-functions will be changed with customization choices. For example, the sub-function play back music is expected to be standard on any iPod. Without this the product would essentially be an expensive hard drive. An example of a sub-function that allows for better functionality control would be the play back video option. This sub-function is discussed in greater detail in the case study demonstration section. The main sub-functions that are to be customized are ones that include some surprise and delight features. The Kano model [34] discusses products having must have (critical to quality and function), linear (would prefer more or less of a feature) and surprise and delight features (features that set the product apart). The presented approach to configurable design does not purpose that the consumer customize the “must have” features. In most cases the consumer will be customizing the “linear” or “surprise and delight” features on a product.

4.3 Case Study Demonstration

The first question is selected to demonstrate the approach. This question is primarily related to component 1 (LCD screen). However it does concern four of the eight functions in the FCM. The first component in the FCM has two listed level 2 dependencies. The two sub-functions are: play back video, and display information. For instance without video play back the LCD screen component is no longer required. Without the LCD screen, the iPod can not play back video or display any

information either, hence the level 2 dependencies that exist with the two sub-functions. This is shown in Table 6.

Functions	LCD Screen
play back music	1
play back video	2
store data	0
receive energy	0
store energy	0
display information	2
receive user input	1
sync	0

Table 6 - LCD Screen Dependencies to All Sub-functions

The numbers in the system have been mapped to three colors, red = 2, yellow = 1, green = 0 for use in the program. The colors are used in the text prompts to the customer as they make customization choices. The colors are selected with the intent of showing the severity of the choice being made. The assumption is that the color red is an undesirable choice while green is a more desirable choice. This color scheme is used in the interface while prompting the user.

If the user decides not to have an LCD screen by stating they do not wish to watch video on the iPod, then the play back video, and display information sub-functions are affected the most. The algorithm takes out the LCD screen component from the iPod. The other sub-functions are also checked to determine the impact of this choice. Based on this it will prompt the user accordingly.

Table 7 shows the logic flow of the algorithm in the interface. The process takes the following form:

- Preference is stated through a customization question relating to a sub-function.

- If a sub-function is removed the algorithm will check every column of that sub-function to see if other dependencies exist.
- If other dependencies exist the user is informed of the consequences of the preference choice. This is represented with the arrows going down each column that has a dependency greater than 0 for F2.

Figure 4 shows what the user sees as they make this choice. The interface has gone through the logic flow shown in Table 7 and now informs the user of the other sub-functions that are affected.

Sub-Function	LCD Screen	Video Decoder	Audio Decoder	Hold Button
F1: Play back audio	↑ 1	↑ 0	↑ 2	0
F2: Play back video	2	2	2	0
F6: Display	2	2	0	0
F7: Receive user Input	↓ 1	↓ 2	↓ 0	0

Table 7 - Algorithm Logic Flow

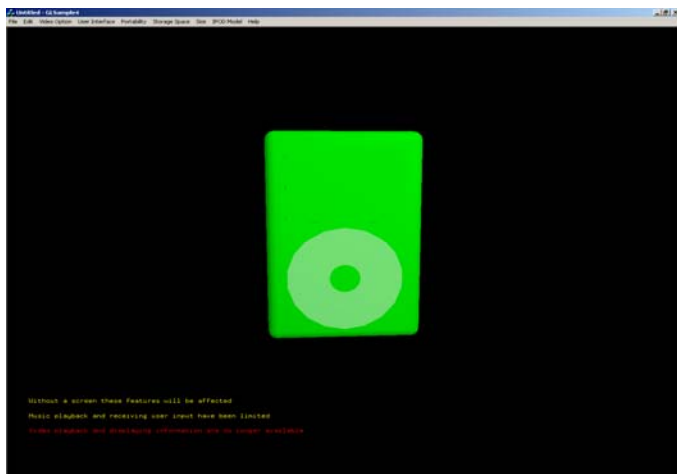


Figure 4 - Interface Reflecting a ‘No’ Answer to Question 1

Depending on the customization choices being made, the FCM may need to be updated to reflect the individual’s preference structure. If the user answers question 1 with a ‘No’, indicating their preference to not have an LCD screen on the iPod, then an update on the FCM is not required. If the answer to this question is ‘Yes’, then the FCM is updated to reflect this choice. Figure 5 shows what the user sees as they make this choice. The preference has made the LCD screen a required component in the iPod. By choosing to watch video on the iPod the four functions that are associated with this choice, are not limited in any way.

Table 8 shows the affected entries in the FCM based on a ‘Yes’ answer to question 1. Only three of the eleven components within the functions have been affected based on this question. The three components are LCD Screen, Video decoder, and

Audio decoder. The program goes through the columns and replaces the entries all with a level 0 (no functionality limitation) entry. Having replaced the level 1 and 2 dependencies with a level 0 dependency lets the interface know that these particular areas have no limitations to their functionality based on the customer’s preference. This allows for further customization with these areas not conflicting with the process.

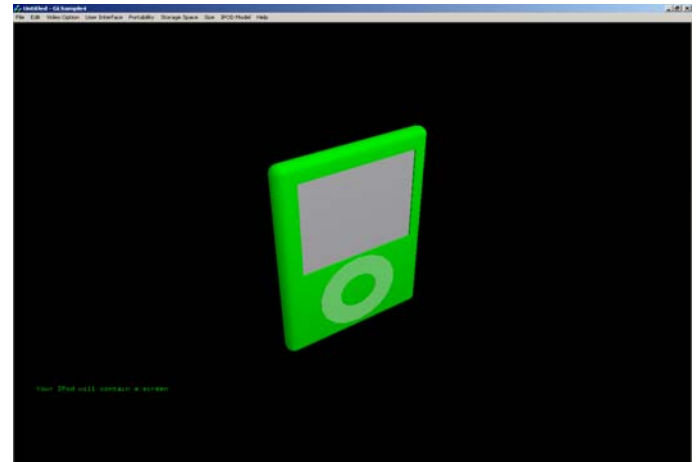


Figure 5 - Interface Reflecting a ‘Yes’ Answer to Question 1

Functions	LCD Screen	Video decoder	Audio decoder
F1 Play back music	0	0	0
F2 Play back video	0	0	0
F6 Display information	0	0	0
F7 Receive user input	0	0	0

Table 8 - Updated FCM Reflecting a Yes Answer to Question 1

The algorithm also makes sure that any choices that are being made do not conflict with each other. Sometimes a consumer may have inconsistent preferences while customizing a product. The interface tracks such inconsistencies and alerts the user if any exist during the customization process. As an example, if a user states that they do not wish to watch videos on the iPod, meaning that the iPod would not have a screen. If for any reason later in the customization process the consumer makes a selection that requires a screen on the iPod, the interface would let them know that they have already stated that they did not want a screen and that this component was removed from the iPod.

This preference inconsistency is shown in Figure 6. Here the user has stated that they do not wish to watch video on the iPod, yet they want the iPod to have a touch screen. The interface lets the user know that these two preferences are not consistent. Watching video and having a touch screen both require a screen. Having one of the options automatically

provides the other. As stated in Section 3.3, adding one component through a given sub-function may provide other sub-functions which the customer has not expressed a preference towards. This preference inconsistency needs to be resolved before the product can be finalized. If this error is not resolved the feasibility check would catch the error and inform the user again to switch one of the two preferences that are not consistent.

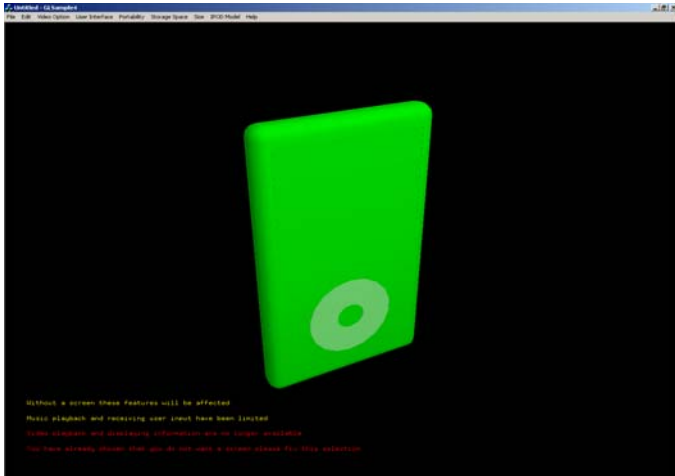


Figure 6 - Inconsistent Preference Structure

The customer can explore the options as long as they wish and make multiple changes to the same part of the iPod. Once customizing the product is complete the finalize line command is activated. The feasibility check runs through the FCM making sure that all the choices are consistent and that the selections are compatible with each other. This is the final part of the customization process and at this point, the customer would be ready to place an order.

5. CONCLUSIONS

Using the iPod as a case study we have demonstrated that FCM's can be used in a customization process to aid both the engineer and the customer. By using FCM's not only does a design engineer gain an understanding of where customization can take place on a product, but they can plan ahead and foresee the change propagation that would take place in a product due to any changes that customers may make. This allows a designer to create a controlled functional customization interface that can be used over the web to sell products. This is also the first example of using a FCM interface with the customer to customize a product. Giving the consumer this level of control over a product design allows for a functional customizable product and ensures functionality of the product.

Using a tri-level rating scheme in the FCM can limit the relationship level from each component to each sub-function.

If the overall function of the product is more complicated, a larger range of levels may be needed in order to capture all the relationship information required to create a customization interface, however this area requires more research. Also, the ability for a designer to construct good questions for the customer to customize sub-function can be a limiting factor in this method. If the questions do not relate well to the sub-functions or are hard to understand, the effectiveness of the method can be reduced.

Future work will focus on generalization of this method to be able to work with any product or as many products as possible along with advancing the feasibility check to include a geometric checker to be sure that all selected components will fit together in one case.

6. ACKNOWLEDGEMENTS

We are grateful to the New York State Center of Engineering Design and Industrial Innovation (NYSCEDI) through support from the New York State Office of Science, Technology, and Academic Research, Grant #M050100 for providing computational support for the project. We also acknowledge funding from the Department of Education, Fund for the Improvement of Postsecondary Education, Grant #P116B060438.

7. REFERENCES

1. Gilmore, J. and Pine, B.J., 2000, Markets of One: Creating Customer-Unique Value through Mass Customization, Harvard Business School Publishing.
2. Zeleny, M., 1997, "Towards the Tradeoffs-Free Optimality in MCDM," In *Multicriteria Analysis*, edited by J. Climaco (Berlin, Heidelberg: Springer-Verlag), pp. 596-601.
3. Shiau, C., Tseng, I., Heutchy, A., Michalek, J., 2007, "Design Optimization of a Laptop Computer Using Aggregate and Mixed Logit Demand Models with Consumer Survey Data" *ASME International Design Technical Conferences*, DETC2007-34883, Las Vegas, NV.
4. Otto, K., and K. Wood, 2001, Product Design: Techniques in Reverse Engineering and New Product Development, Prentice Hall, Upper Saddle River, New Jersey.
5. Shooter, S. B., Simpson, T. W., Kumara, S. R. T., Stone, R. B. and Terpenney, J. P. 2005, "Toward a Multi-Agent Information Infrastructure for Product Family Planning and Mass Customization," *International Journal of Mass Customization*, 1(1), pp. 134-155.
6. Hirtz, J., Stone, R., McAdams, D., Szykman, S. and Wood, K., 2002, "A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts," *Research in Engineering Design*, 13(2), pp. 65-82.

7. Greer, J., Stock, M., Stone, R. and Wood, K., 2003, "Enumerating the Component Space: First Steps Toward a Design Naming Convention for Mechanical Parts," *ASME International Design Technical Conferences*, DETC2003/DTM-48666, Chicago, IL.
8. Bryant, C.R., Stone, R.B., McAdams, D.A., Kurtoglu, T., Campbell, M.I., 2005, "Concept Generation from the Functional Basis of Design," *International Conference on Engineering Design*, Melbourne, Australia.
9. Pahl, G. and Beitz, W., 1996, Engineering Design: A Systematic Approach, Springer-Verlag.
10. Jones, J., 1987, Design Methods, David Fulton Publishers, London.
11. Ullman, D., 1997, The Mechanical Design Process, McGraw-Hill.
12. Szykman, S., J. Racz and R. Sriram, 1999, "The Representation of Function in Computer-Based Design," *ASME International Design Technical Conferences*, DETC99/DTM-8742, Las Vegas, NV.
13. Koch, P., Peplinski, J., Allen, J. and Mistree, F., 1994, "A Method for Design Using Available Assets: Identifying a Feasible System Configuration," *Behavioral Science*, 30, pp. 229-250.
14. Kirschman, C. and Fadel, G., 1998, "Classifying Functions for Mechanical Design," *Journal of Mechanical Design*, 120(3), pp. 475-482.
15. Bryant, C., Stone, R., McAdams, D., Kurtoglu, T., Campbell, M., 2005, "A Computational Technique for Concept Generation," *ASME International Design Technical Conferences*, DETC05/DTM-85323, Long Beach, CA.
16. Attaluri, V., McAdams, D., Stone, R. and De Crescenzo, A., 2006, "Visual Representations as an Aid to Concept Generation," *ASME International Design Technical Conferences*, DETC2006-99572, Philadelphia, PA.
17. Vucovich, J., Bhardwaj, N., Hoi-Hei, H., Ramakrishna, M., Thakur, M. and Stone, R., 2006, "Concept Generation Algorithms for Repository-Based Early Design," *ASME International Design Technical Conferences*, DETC2006-99466, Philadelphia, PA.
18. Bohm, M., and Stone, R., 2004, "Representing Functionality to Support Reuse: Conceptual and Supporting Functions," *ASME International Design Technical Conferences*, DETC2004-57693, Salt Lake City, UT.
19. Tumer, I. and Stone, R., 2003, "Mapping Function to Failure During High-Risk Component Development," *Research in Engineering Design*, 14(1), pp. 25-33.
20. Bohm, M., Stone, R. and Szykman, S., 2003, "Enhancing Virtual Product Representations for Advanced Design Repository Systems," *Proceedings of DETC2003*, DETC2003/CIE-48239, Chicago, IL.
21. Strawbridge, B., McAdams, D. and Stone, R., 2002, "A Computational Approach to Conceptual Design," *ASME International Design Technical Conferences*, DETC2002/DTM-34001, Montreal, Canada.
22. McAdams, D., Stone, R., and Wood, K., 1999, "Functional Interdependence and Product Similarity based on Customer Needs," *Research in Engineering Design*, 11(1), pp. 1-19.
23. Stone, R., Wood, K., and Crawford, R., 2000, "Using Quantitative Functional Models to Develop Product Architectures," *Design Studies*, 21(3), pp. 239-260.
24. Ferguson, S., Siddiqi, A., Lewis, K. de Weck, O., 2007, "Flexible and Reconfigurable System: Nomenclature and Review," *ASME Design Engineering Technical Conferences*, Las Vegas, NV, DETC2007-35745.
25. Singh, S., Skiles, V., Krager, J., Wood, K., Jensen, D., Szmerekovsky, 2006, "Innovations in Design Through Transformation: A Fundamental Study of Transformation Principles," *ASME Design Engineering Technical Conferences*, Philadelphia, PA, DETC2006-99575.
26. Puma Corporation, <http://mongolianshoebq.puma.com/pindex.jsp>, (accessed December 13, 2007).
27. Shooter, S., Simpson, T., Kumara, S., Stone, R. and Terpenney, J., 2004, "Toward an Information Management Infrastructure for Product Family Planning and Platform Customization," *ASME International Design Technical Conferences*, DETC2004-57430, Salt Lake City, UT.
28. Simpson, T. W., 2004, "Product Platform Design and Customization: Status and Promise," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18(1), pp. 3-20.
29. Simpson, T. W., Marion, T. J., de Weck, O., Holtta-Otto, K., Kokkolaras, M. and Shooter, S. B., 2006, "Platform-Based Design and Development: Current Trends and Needs in Industry," *ASME Design Engineering Technical Conferences*, DETC2006/DAC-99229, Philadelphia, PA.
30. Simpson, T. W., Nanda, J., Sachin, H., Umopathy, K., and Hodge, B., 2003, "Development of a Framework for Web-based Product Platform Customization," *ASME Journal of Computing and Information Science in Engineering*, 3(2), pp. 119-129.
31. Huffman, C. and Kahn, B. E., 1998, "Variety for Sale: Mass Customization or Mass Confusion," *Journal of Retailing*, 74(4), pp. 491-513.
32. Mather, H., 1995, "Product Variety -- Friend or Foe?" *Proceedings of the 1995 38th American Production & Inventory Control Society International Conference and Exhibition*, Orlando, FL, APICS, pp. 378-381.
33. Stalk, G., Jr. and Webber, A. M., 1993, "Japan's Dark Side of Time," *Harvard Business Review*, 71(4), pp. 93-102.
34. University of Calgary, <http://www.ucalgary.ca/~design/engg251/FirstYearFiles/kano.pdf> (accessed January 11, 2008).