Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 48 (2016) 271 – 276

www.elsevier.com/locate/procedia

23rd CIRP Conference on Life Cycle Engineering

Dynamic modular architecture for product lifecycle

Shraga Shoval, Li Qiao, Mahmoud Efatmaneshnik and Michael Ryan

Capability Systems Centre, School of Engineering and Information Technology (SEIT)
University of New South Wales Canberra
CANBERRA 2610, Australia

Corresponding author. Tel.: +61 (2) 6268 9566; E-mail address: s.shoval@adfa.edu.au

Abstract

A module is a set of components with interfaces selected in order to help designers address ilities or non-functional system requirements. Consequently, the boundaries of a module do not necessarily coincide with those dictated by functional decomposition. Modularization usually makes the architecture more complex due to additional interfaces and redundancies that have negative consequences on system performance. As a result, modularization is accompanied by a trade-off between non-functional and functional requirements. Additionally the system lifecycle consists of several phases, each characterized by different activities and goals. Systems may benefit from different modular architectures in the different lifecycle phases. This paper presents a dynamic modular architecture methodology, where the modular architecture changes over the different product lifecycle phases. An example of a relatively simple mechanical system - a bicycle – is presented to illustrate the implementation of the methodology.

© 2016 The Authors. Published by Elsevier B.V This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of the scientific committee of the 23rd CIRP Conference on Life Cycle Engineering

Keywords: Modularization; System lifecycle; Design Structure Matrix; Clustering;

1. Introduction

Decomposition of a system into lower-level structures based on its functional requirements is a common tool in systems design and development, especially for complex systems. Decomposition and modularization are not necessarily identical [1], and the major difference between the two terms is that a system's functional decomposition is often related to engineering requirements, while modularization is undertaken in order to accommodate the desired system ilities, which are defined as non-functional requirements. Ilities are used for appraising the entire system at specific phases and from a specific viewpoint over the system's lifecycle, rather than satisfying the functional requirements of specific elements in the system. Ilities usually add beneficial or even luxurious features to the essential functional requirement of the system. As a result, the functional requirements affect the system's functional structure, and ilities are accommodated by the system's modular architecture. Modular design is commonly implemented as part of the system design [2,3,4] for creating architecture that exhibits ilities, in addition to the required operational features.

Many methods have been suggested for system modularization. The Group Technology (GT) methods [5,6,7,8] group products, machines, tools, and manufacturing processes into manufacturing families (cells). The GT methods are based on common features of the products (e.g. geometries, materials, shapes) and their manufacturing processes. The outcome of the GT methods is a *cellular manufacturing system* that results in optimal flow and increased efficiency. A liaison network that represents the functional, as well as non-functional relations between system's components, is another common tool in the design of modular architectures [9, 10]. Liaison networks are based on nodes that represent the system's elements and arcs that represent the level of coupling and the structural properties

2212-8271 © 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of the scientific committee of the 23rd CIRP Conference on Life Cycle Engineering
doi:10.1016/j.procir.2016.03.037

between the system's elements [11]. While liaison networks have limited capabilities in identifying indirect relations of complex system elements [12], Design Structure Matrices (DSMs) and Multiple Domain Matrices (MDMs) can identify these hidden structures in complex systems. Various clustering algorithms are used in DSMs and MDMs in order to group elements into modules, using cost functions which are based on the system's objectives.

Each phase in the system's lifecycle is characterized by different functional requirements, as well as non-functional ilities [13,14]. Unfortunately, modularity and functionality don't always have a positive relationship. In fact, modularity may have adverse effects on functionality as illustrated in Fig. 1. The system's elements are initially divided into three clusters (subsystems) according to functional and logical decomposition at the operation phase. The elements within each cluster have strong connections between them (shown by the solid connection lines) with looser or no connections with elements in the other subsystems (shown by the thinner connecting lines). Ideally, the connections between clusters are weak such that they can be easily disconnected from other clusters. However, the modular architecture that is based on the system's ilities may constitute a different structure that, in some instances, may contradict the functional structure during the operation phase.

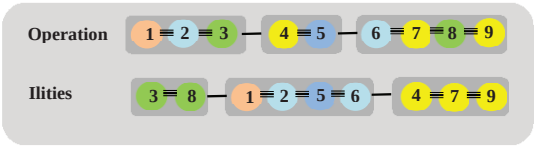


Fig. 1. Changes of modular configuration in system's lifecycle

In this paper we discuss the design of system modularization given the differences between the system's functional structure and the architecture driven by the system's ilities. A dynamic modularization concept is proposed. The dynamic modularization considers the system's structure based on functional requirements and non-functional illities. Newcomb et al. [15] present two measures for analysing modular architectures in what they call "lifecycle viewpoints." The measures consider the differences between the components of various modules and the interdependencies between the clusters. Modularity is determined by the weighted sum of the two measures. Alizon et al. [16] analyse product family design in terms of the uniqueness, varieties, and commonalities of modules in family products. The objective is to determine common features among all modules. These features can then be used in a large range of products from the same family.

The concept proposed in this paper considers the variance between the modular structures resulted from functional and non-functional considerations through the lifecycle phases. The outcome of the proposed model is a numerical value that indicates the clustering cost of a particular architecture, given the interdependencies of the elements in the functional structure and the modularization requirements as derived from the system's ilities.

2. Problem formulation

Consider a system that consists of n elements. A square matrix $M_F \in (n \times n)$ also known as the functional DSM (we refer to this matrix as DSM^F), that maps the interactions of the system's elements according to the functional requirements. A numerical value $m_F(i,j)$ represents the level of interaction between element i and element j (larger values represent stronger interaction or dependency between the elements). A similar procedure is performed for the construction of the system's ilities DSM – $M_I \in (n \times n)$ (we refer to this matrix as DSM^I) that consists of the same n elements as in the functional DSM^F. Once the two DSMs are constructed, a clustering procedure is performed independently on each matrix to generate groups of elements that have strong internal interactions, and minimal or no interactions and dependencies between the groups. As mentioned, many clustering algorithms have been proposed over the past 50 years. Early clustering algorithms originally developed for Group Technology use matrix permutations of rows and columns. Other algorithms use techniques based on versatile objective functions. Zakarian [17] proposes a model for non-binary as well as binary matrices that considers interactions that are categorized as "bottlenecks". Bottleneck interactions are outside the clusters, representing interactions or dependencies between elements that belong to different clusters. The objective is to minimize the weights of the bottlenecked interactions by reorganizing the elements related to these bottlenecks. The efficiency of the clustering process is measured by the summation of all the bottleneck interactions. The algorithm starts with n clusters, each containing a single element (with maximal bottleneck interactions) resulting in the worst clustering cost. As the algorithm proceeds, the number of clusters is reduced and the dimension of the clusters (number of elements within the clusters) increases. The algorithm continues to extend the clusters until no bottleneck interactions remain. In an "ideal" system, the outcomes are clusters with closely related elements and no bottleneck connections. If no "ideal" clustering is found, the algorithm continues until eventually it constructs one large cluster that consists of all elements in the system. While mathematically the latter case is acceptable (as there are no bottleneck interactions and the cost is zero), this is usually not a feasible solution. There is a need to define a modified objective function that considers, in addition to the the bottleneck interfaces, the number of the clusters and their size. For example, consider the following objective function:

$$C_{tot} = \sum_{i=1}^{k} C_{int}(i) + C_{ext} \quad (1)$$

$$C_{int}(i) = (W(u,v) + W(v,u)) * d_i^p \quad (2)$$

$$C_{ext} = (\sum(B(u,v) + B(v,u))) * n^p \quad (3)$$

where C_{tot} - total clustering cost
$W(u,v)$ - internal connection within a cluster
$B(u,v)$ - external connection (bottleneck connections)
$C_{int}(i)$ - internal cost of cluster i
C_{ext} - external cost
d_i - dimension (size) of cluster i
n - number of elements in the entire system
p - penalty factor

The designers may imply constraints regarding the total number of clusters, the clusters' size, and/or the bottleneck connections. To illustrate, consider the bicycle shown in Fig. 2 that is comprised of 21 parts. The actual number of components is much larger (around 1000 parts), but for clarity and simplicity we show only the major parts, listed in Table 1. Our objective is to cluster the bicycle's elements into groups according to their functional requirements.

Table 1: Bicycle components

No.	Name	No.	Name	No.	Name
1	Saddle	8	Fork	15	Crank arm
2	Seat post	9	Front brake	16	Pedal
3	Seat tube	10	Head tube	17	Rear brake
4	Top tube	11	Handlebar	18	Rear derailleur
5	Seat stay	12	Handlebar grip	19	Front derailleur
6	Down tube	13	Front wheel	20	Chain
7	Chain stay	14	Rear wheel	21	Chain ring

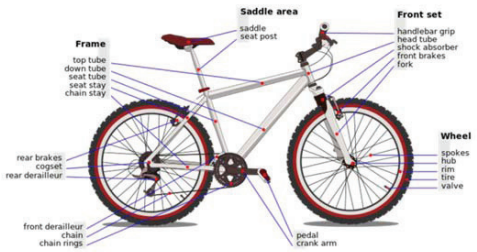
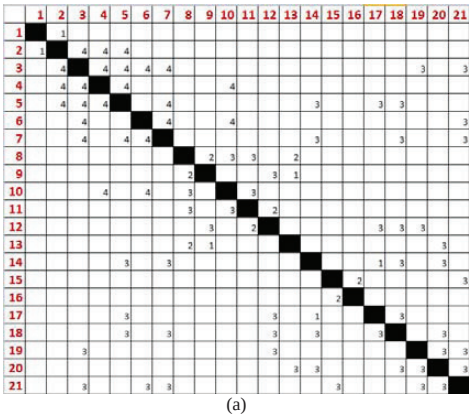
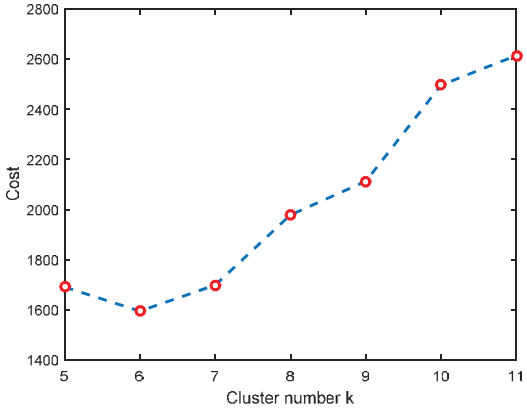


Fig. 2. Typical Bicycle components [Source: http://www.buzzle.com/images/autos/road-bike.jpg]

The DSM shown in Fig. 3(a) represents the functional connectivity of the bicycle components. The numbers within the matrix represent the strength of the connection where 4 represents a critical connection (a break in this connection would be catastrophic for the system's operation) and 1 represents a loose connection (e.g. elements with connections that have a minor effect on the operation of the system and can be easily and quickly replaced). The outcome of the clustering process based on the DSM is a list of possible clusters, each associated with a cost that provides a numerical value according to the objective function (1-3). Fig. 3(b) shows the quality function $Cost$ versus the cluster number k for DSM^F shown in fig. 3.



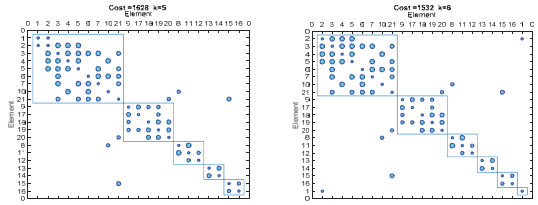
(a)



(b)

Fig. 3. The functional DSM for the 21 components of a bicycle (a), and the clustering cost for different number of clusters.

Figure 4 shows two samples of possible structures with different number of clusters (and costs).



(a) (b)

Fig. 4. Examples for 2 possible structures: 5 clusters with a total cost of 1628 (a) and 6 clusters with a total cost of 1532.

It is clear from Fig. 3(b) that a structure which consists of 6 clusters is optimal in terms of the clustering cost. Table 2 lists the elements in each cluster for this structure. These clusters represent subassemblies/components/mechanisms that have high intra-connectivity of their elements, and looser connectivity between elements among different clusters.

Table 2: Optimal clusters for DSM^F

Cluster 1 (8)	Seat post (2) Seat tube (3) Top tube (4) Seat stay (5) Down tube (6) Chain stay (7) Head tube (10) Chain ring (21)
Cluster 2 (5)	Front brake (9) Rear brake (170 Rear derailleur (18) Front derailleur (19) Chain (20)
Cluster 3 (3)	Fork (8) Handlebar (11) Handlebar grip (12)
Cluster 4 (2)	Crank arm (15) Pedal (16)
Cluster 5 (2)	Front wheel (13) Rear wheel (14)
Cluster 6 (1)	Saddle (1)

3. Clustering for ilities

Next, we repeat the clustering procedure described in section 2 using the system ilities rather than the functional requirements. As mentioned, in an 'ideal' system the clustering outcomes of the DSM^F and the DSM^I form an identical structure. However, in most systems the two procedures result in different structures as described in Fig. 1. The clustering results of the DSM^I are shown in Fig. 5 and Table 3. As shown, the numbers of clusters, as well as their contents are different from those of the DSM^F.

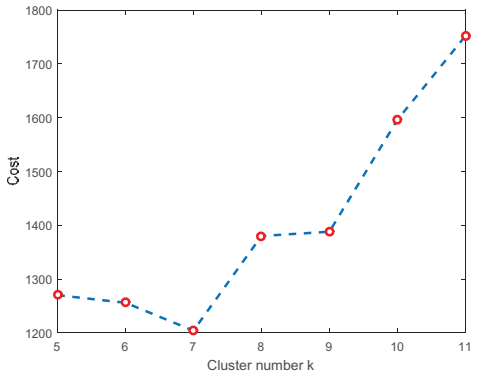


Fig. 5. DSM^{I} clustering structures

Table 3: Optimal clusters for DSM^{I}

Cluster 1 (6)	Seat post (2) Seat tube (3) Top tube (40 Seat stay (5) Down tube (6) Chain stay (7)
Cluster 2 (5)	Crank arm (15) Pedal (16) Front derailleur (19) Chain (20) Chain ring (21)
Cluster 3 (3)	Rear wheel (14) Rear brake (17) Rear derailleur (18)
Cluster 4 (2)	Fork (8) Head tube (10)
Cluster 5 (2)	Front brake (9) Front wheel (13)
Cluster 6 (2)	Handlebar (11) Handlebar grip (12)
Cluster 7 (1)	Saddle (1)

4. Combined clustering structure

Once clustering is completed for both the DSM^{I} and DSM^{F}, a 3D DSM (called DSM^{3}) is constructed [18]. The DSM^{3} consists of two layers: the DSM^{I} and the DSM^{F} (Fig. 6). The goal of DSM^{3} is to find a dynamic structure given the differences between the DSM^{I} and DSM^{F}.

In general there are three possible types of clustering architectures in the DSM^{3}:

Type I – clusters that are identical in both the DSM^{I} and the DSM^{F}. The identity between the clusters is characterized in terms of the dimension (number of elements) and the content (elements within the clusters) for both layers.

Type II – large clusters that consist of a mixture of elements from entire smaller clusters from the other layer of the DSM^{3}.

Type III – clusters that exist in one DSM layer and not in the other. In this case some connections are broken, and new connections are created in the transfer from one DSM layer to the other.

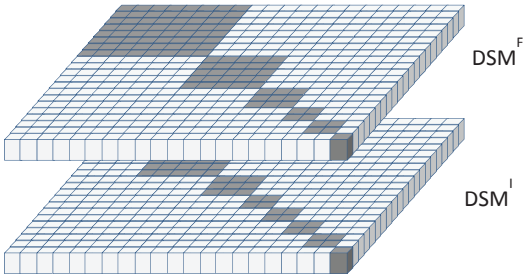


Fig. 6. A two layer DSM^{3} consisting of the DSM^{I} and the DSM^{F}.

In this work we use the *Jaccard similarity index* in order to measure the similarity between clusters from different layers. Given a clustering procedure that results in *m* clusters of the DSM^{F} and *n* clusters in the DSM^{I}, a similarity matrix *S(m×n)* is constructed. The matrix represents the similarity between any two clusters in the two layers of the DSM^{3}. The $s_{k,l}$ element in the similarity matrix is the *Jaccard similarity coefficient* between cluster *k* of DSM^{F} and cluster *l* of DSM^{I} given by

$$s_{(k,l)} = \frac{p}{p+q+r} \quad (4)$$

where

p – number of common elements in the cluster *k* of DSM^{F} and in cluster *l* of DSM^{I}

q – number of elements in cluster *k* of DSM^{F} and not cluster *l* of DSM^{I}

r – number of elements not in cluster *k* of DSM^{F} and in cluster *l* of DSM^{I}

Fig. 7 shows the similarity matrix for the DSM^{3} that consists of the clusters listed in table 2 and 3 (the bicycle example). For example, the value of element $s_{2,4}$ is determined by the content of cluster 2 in the DSM^{I} and cluster 4 in the DSM^{F}. There are 2 common elements in both clusters (15 and 16) and 3 elements in DSM^{I} that are not in the DSM^{F} (19, 20 and 21) resulting a similarity coefficient of 2/5.

	M_1^F	M_2^F	M_3^F	M_4^F	M_5^F	M_6^F
M_1^I	$^6/_8$	$^1/_{12}$	0	0	0	0
M_2^I	$^1/_{12}$	$^2/_8$	0	$^2/_5$	0	0
M_3^I	0	$^2/_6$	0	0	$^1/_4$	0
M_4^I	$^1/_{98}$	0	$^1/_4$	0	0	0
M_5^I	0	$^1/_6$	0	0	$^1/_3$	0
M_6^I	0	0	$^2/_3$	0	0	0
M_7^I	0	0	0	0	0	1

Fig. 7. The similarity matrix for the clusters in Tables 2 and 3.

5. Cost estimate of combined clustering

In this section we present the estimate of the costs associated with accommodating the different requirements and ilities clusters through the system's lifecycle phases as expressed in the DSM^{F} and the DSM^{I}.

5.1. Type I clusters

Since clusters of *Type I* are identical in both DSM^{F} and DSM^{I}, there is no need to modify their content or the connectivity of the elements in these clusters. These clusters are characterized by a value of '1' in the similarity matrix, indicating that the two clusters have 100% similarity according to the *Jaccard similarity coefficient*. No costs are associated with the transition of these clusters.

5.2. Type II clusters

Type II clusters refer to cases where elements of a cluster or a group of clusters are entirely contained within a larger

cluster in the other layer. The clusters of *Type II* are characterized in the similarity matrix by a row or column that consists of a single non-zero value. In the similarity matrix of Fig. 6, cluster M_4^F has a single non-zero index with cluster M_2^I, indicating that it is contained entirely within the larger cluster. Similarly, cluster M_6^I is entirely contained within cluster M_3^F. This type of architecture has two costs associated with it: the disconnect cost between the smaller clusters and the neighbouring clusters, which ideally are weak, and the reconnecting cost of the elements in the larger cluster.

5.3. Type III clusters

In this type, elements from a cluster in one layer of the DSM^3 are transferred to another cluster in the other layer. There are two costs associated with each element's transition: the disconnection cost of leaving the current cluster, and the connection cost for joining the elements in the new cluster.

6. A clustering method for cost reduction based on the similarity matrix.

Consider a cell $s_{(k,j)}$ in the similarity matrix *S* that represents the *Jaccard coefficient* between clusters *k* and *j* in the DSM^F and DSM^I respectively. Scanning through all rows in *S*, we first search for similarity coefficients with the value of '1' (representing *Type 1* clusters). Row *k* and column *j* of these clusters can be deleted without affecting other clusters as all coefficients in the k^{th} column and j^{th} row are '0' except for $s_{(k,j)}$. As discussed, common clusters have no additional cost when being shifted between the layers of the DSM^3.

Next, we look for *Type II* clusters. *Type II* clusters have a single non-zero value in their row or column. This indicates that the cluster is fully contained within a single larger cluster. The additional costs are associated with disconnecting the cluster from other clusters in one layer and reconnecting its elements within the larger cluster in the other layer. Once the costs are determined, the rows and columns of the two clusters are combined in the updated similarity matrix. A new 'super-cluster' is added to the similarity matrix instead of the two old clusters with the re-clustering cost associated with it.

Finally, we look for *Type III* clusters. There are two parameters – *q* and *r* that affect the cost of the re-clustering process. *q* refers to all elements from the old cluster that are to be disconnected, and *r* refers to all elements in the new clusters that need to be connected. The connection and disconnection costs are expressed in DSM^F and DSM^I, assuming these costs are proportional to the strength of connectivity between the elements. We start by identifying the clusters with the highest similarity values. The rationale is that these clusters require fewer changes, in terms of elements and connections, relative to other possible pairs. If there are several cells with equal values in the row (or column), then any of them can be selected randomly. Once elements are transferred from one cluster to the other, a new similarity matrix is constructed with the updated clusters, and the process is repeated. At each stage the cost of re-clustering is calculated and attached to the updated clusters. As with the clustering algorithm in a single layer DSM, the process can continue until an 'ideal' solution is achieved (all elements in the clusters of one layer are transferred to clusters in the other layer), or until all clusters of *Type III* are combined into a single large cluster in the other layer. The system's designer can determine the constraints regarding the acceptable number of clusters and their dimensions.

Consider the similarity matrix shown in Fig. 6; there is one cluster of *Type I* - $M_6^F = M_7^I$ and therefore there is no cost associated with it**.** Also, there are two pairs of cluster of *Type II:* $\{M_4^F - M_2^I\}$ and $\{M_6^I - M_3^F\}$. There are no additional costs in combining cluster M_4^F with clusterM_2^I as all connections are maintained through DSM^F and DSM^I. However, an additional cost is required when combining clusters M_6^I and M_3^F. This cost is related to elements 11 (Handlebar) and 8 (Fork). The strong connectivity between these two elements ('3' in DSM^F) is replaced by a weak connectivity ('1' in DSM^I) resulting in a cost of 2. In conventional bicycle design, the connection between the Fork and the Handlebar is performed by the 'Stem' mechanism which consists of two types: **Quill type stem** that is inserted into a threaded Head tube, and **Thread-less type stem** that clamps around the Head tube. The Thread-less stem is an example of a modified design that, although requiring additional cost, provides ilities using a modular design. It offers less labor and simpler way to disconnect, swap, flip, mix and match, requiring a simple Allen wrench. It provides the strong connectivity required by the DSM^F, and a simple quick release mechanism required by the DSM^I. A potential user can evaluate the additional cost vs. the upgraded ilities when comparing the two models for the given functional and ilities requirements. Fig. 8 shows the two types of Stem mechanisms in bicycles.



Fig. 8. Quill (a); and Thread-less stems.

Fig. 9 shows the updated similarity matrix after eliminating *Type I* and combining *Type II* clusters. The combined clusters are registered with {*}, and their similarity values are updated. For example, the combined cluster $\{M_2^I + M_4^F\}^*$ consists of elements {15, 16, 19, 20 and 21} and it has a similarity coefficient of $^2/_8$ with cluster M_2^F based on 2 common elements (19 and 20) out of the total 8 elements. Similarly, the combined cluster $\{M_3^F + M_6^I\}^*$ has a similarity coefficient of $^1/_4$ with cluster M_4^I where element 8 is common and elements {10, 11, and 12} are not. Further potential upgrades are based on the connectivity of components in clusters M_1^F and M_1^I with a similarity value of 0.75, clusters M_2^F and M_3^I, and clusters M_5^F and M_5^I, both with similarity values of $^1/_3$.

	M_1^F	M_2^F	$\{M_3^F + M_6^I\}^*$	M_5^F
M_1^I	$^6/_8$	$^1/_{12}$	0	0
$\{M_2^I + M_4^F\}^*$	$^1/_{12}$	$^2/_8$	0	0
M_3^I	0	$^2/_6$	0	$^1/_4$
M_4^I	$^1/_8$	0	$^1/_4$	0
M_5^I	0	$^1/_6$	0	$^1/_3$

Fig. 9. Updated similarity matrix after considering *Type I* and *Type II* clusters.

The elements that are 'candidates' for the next updates are the Rear wheel (14), Rear derailleur (18) and the Chain ring (21). Examining various bicycle models shows that upgraded models, which are usually more expensive, provide a more modular design for these elements. For example, the upgraded internal hub-gearing mechanism, shown in Fig. 10a replaces the traditional derailleur gearing system (Fig. 10b). Although significantly more expensive and heavier than the derailleur gearing system, the internal hub gearing is simpler for regular maintenance, in particular when frequent assembly and disassembly of the rear wheel is required. Again, the user can evaluate the additional cost and compare the ilities associated with internal gearing with the conventional derailleur mechanism.



(a) *(b)*

Fig. 10. Internal hub gearing mechanism (a); and conventional derailleur gearing mechanism (b)

7. **Conclusions**

This paper analyses the effect of changes in a system's clustering structure during its lifecycle. These changes may be due to operational requirements, and to the system's ilities. The paper proposes a model for determining these additional costs using a multi-layer Data Structure Matrix (DSM), called DSM^3. The DSM^3 consists of the functional DSM, named DSM^F and the ilities DSM named DSM^I. The additional costs are associated with reconfiguring clusters by detaching elements from their initial cluster, and re-connecting them to new clusters according to subsequent lifecycle phases. The analysis is based on similarity matrices that determine the differences between the clusters' configurations through the system's lifecycle phases. The proposed methodology is illustrated using a simple mechanical product – a bicycle. Bicycles in their modern formation have existed for over 200 years, with a wide range of sizes, shapes and prices. While maintaining their basic functional formation for over 180 years, recent design modifications offer ilities that add features to the basic functional requirements. Some modified features are becoming part of the standard design of conventional models, but other beneficial features are considered luxurious, and therefore are found only in more lavish models.

The model proposed in the paper can assist systems engineers during the preliminary design phase of a system's lifecycle in combining functional requirements and non-functional ilities. A successful integration of functional requirements and ilities produces improved designs that benefit all system's stakeholders.

References

[1] Kusiak A., and Huang C. C., "Development of modular products," IEEE Trans. on Manufacturing Technol., Vol. 19, No. 4, pp. 523–538, 1996.

[2] Zakarian A., and Rushton G. J., "Development of modular electrical systems," IEEE/ASME Transactions on Mechatronics, Vol. 6, no. 4, pp. 507–520, 2001.

[3] Pimmler T.U., and Eppinger S.D., "Integration analysis of product decomposition," Proceeding of ASME Design Theory Methodologies Conference, vol. 68, pp. 342–351, 1994.

[4] A. Kusiak and C. C. Huang, "Modularity in design of products and systems," IEEE Transactions on Man, Systems and Cybernetics, A, vol. 28, no. 1, pp. 66–77, 1998.

[5] McCormick, W. T., Schweitzer P. J.,. White T. W., "Problem Decomposition and Data Reorganization by a Clustering Technique", Operations Research, Vol. 20, pp 993-1009, 1972.

[6] King, J. R., "Machine component grouping in production flow analysis: An approach using rank order clustering algorithm," International Journal of Production Research, vol. 18, pp. 213–232, 1980.

[7] Heragu S. S., "Group technology and cellular manufacturing," IEEE Trans. on Systems, Man and Cybernetics., vol. 24, no. 2, pp. 203–214, 1994.

[8] Shu M.,"Bond energy, rectilinear distance and worse-case bound for the group technology problem," Journal of Operational Research, Vol. 42, No. 7, pp. 571–578, 1991.

[9] Rivkin J. W., Sigelkow N., " Patterned interactions in complex systems: implications for exploration", Management Science, 53 (7) (2007), pp. 1068–1085

[10] Barabasi A., "Scale-free networks: a decade and beyond", Science, 325 (2009), pp. 412–413

[11] Braha D., Bar-Yam Y., "The statistical mechanics of complex product development: empirical and analytical results", Management Science, 53 (7) (2007), pp. 1127–1145.

[12] Baldwin, C ; Maccormack, A ; Rusnak, J., "Hidden structure: Using network methods to map system architecture", Research Policy, 2014 Oct, Vol.43(8), pp.1381-1397

[13] MacCormack A., Baldwin C., Rusnak J., " Exploring the duality between product and organizational architectures: a test of the mirroring hypothesis", Research Policy, 41 (8) (2012), pp. 1309–132

[14] Schoettl F., Lindemann, U., "Design for system lifecycle properties - A generic approach for modularizing systems", Procedia Computer Science, 2014, Vol.28, pp.682-691.

[15] Newcomb P. J., Bras B., Rosen D. w., "Implications of Modularity on Product Design for the Life Cycle", Journal of Mechanical Design, Transactions of the ASME, 1998, Vol. 120(3), pp. 483-490.

[16] Alizon F., Shooter S. B., Moon S. K., Simpson T. W., **"**Three dimensional design structure matrix with cross-module and cross-interface analyses**",** 2007 Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering, DETC2007, 2008, Vol.6, pp.941-948.

[17] Zakarian, A., "A new matrix clustering algorithm for development of system architecture", IEEE Transactions on Systems, man and cybernetics – Part C, Vol. 38, No. 1, 2008, pp. 135-141.

[18] Shoval S., "Dynamic Modularization throughout System Lifecycle Using Multilayer Design Structure Matrices," proceeding of the 13th Global CIRP Conference on Sustainable Manufacturing, September 2015, Ho Chi Minh City/Binh Duong, Vietnam.