Journal of Technology Management & Innovation

E-ISSN: 0718-2724

ljimenez@jotmi.org

Universidad Alberto Hurtado

Chile

Agrawal, Anupam

Product Networks, Component Modularity and Sourcing

Journal of Technology Management & Innovation, vol. 4, núm. 1, 2009, pp. 59-81

Universidad Alberto Hurtado

Santiago, Chile

Journal of Technology Management & Innovation

Received October 23, 2008 / Accepted March 2, 2009 J. Technol. Manag. Innov. 2009, Volume 4, Issue 1

Product Networks, Component Modularity and Sourcing

Anupam Agrawal (1)

Abstract

This paper develops product representations as component networks that evolve from sharing of interfaces with other components in a product and links them to the external world of sourcing. The paper formally defines and develops two measures of component modularity by linking Graph Theory and Product Architecture principles. The first measure, degree modularity, is related to the strength of design dependencies with adjacent components. The second measure, bridge modularity, is related to the criticality of components. These two component modularity measures are calculated and interpreted by studying the actual product architecture of two products - a small machinery product and an automobile subsystem. A sourcing framework is suggested, treating product obsolescence as a moderating variable in the effect of modularity on sourcing. The paper concludes with a discussion of how component modularity measures can help managers to take better decisions in the arena of sourcing – both at an operational level and at the strategic level. Directions for future work are discussed.

Keywords: Product Network; Component Modularity; Sourcing.

(1) Business Administration Department. University of Illinois at Urbana Champaign. Illinois, USA
Email : anupam@illinois.edu. Address: 363, Wohlers hall, S sixth street, Champaign. Phone: +1-217-265-0654

1. Introduction

Complex products are typically considered as a network of components that share interfaces in order to function as a whole (Ulrich, 1995; Suh 2001). Two studies provide the inspiration for the current paper. Herbert Simon posits in his delightful book (Simon,1981) the need for design to balance the internal and external environments (like a clock on a ship – which shows exact time in spite of the pitching and rolling of the ship in a storm). Recent literature (Gershenson et al, 2004) has highlighted the present inconsistencies in the field of modular product design and put forward some critical questions. The current paper is focused on developing and analyzing the stream of research as recommended by Gershenson et al and in the spirit noted by Simon. The paper develops product representations as component networks that evolve from sharing of interfaces with other components in a product and links them to the external world of sourcing. It draws on the branch of mathematics known as 'Graph Theory' to develop measures that quantify the relative degree of modularity of components in complex products, basing it on the patterns of design interfaces of each component.

What makes the study of component modularity interesting? In established firms, organizational subsystems continuously improve particular technological subsystems (modules). This type of innovation can be linked to a stable environment and established firms excel in it. However, architectural innovation or an innovation between the modules may require an adjustment in the relationship between modules. Henderson and Clark (1990) posited that established firms are notoriously bad at architectural innovations. These firms have difficulty managing the inter dependencies between modules. This usually happens since existing organizational structures – which get developed based on the dominant design of the product that is successful for the focal firm - interfere with architectural innovation. New firms or firms entering an industry without pre-existing, module-specific organizational structures have competitive advantage over existing firms since they can align the organization structure along the paths of architecture needed in the market. Therefore, understanding architectural properties, such as component modularity, is particularly important for established firms at the strategic level.

Component modularity is also important at the operational level because it can provide indication to designers about important parameters relating to component performance metrics, such as design rework, or to sourcing managers about key decisions, such as make or buy (Novak and Eppinger, 2001). Understanding of such parameters can lead to better decisions on operational and strategic parameters.

A key feature of product architecture is the level to which it is modular or integral. In the engineering design field, a large stream of research has focused on methods and rules to map functional models to physical components (Kirschman and Fedel, 1998; Newman, 2001; Suh, 2001; Eckert et al, 2004; Jaratt et al, 2004). However, as Ulrich and Eppinger (2004) have suggested, product architecture study involves mapping of functional elements to physical components as well as the specification of the interfaces among interacting components. Hence it is necessary to have tools for measuring the effect of such interfaces among interacting components.

There have been many studies which suggest how to measure whether a product or a subsystem is integral or modular (see for example Ulrich 1995, Sosa et al 2003; Mikkola and Gassmann 2003; Sharman and Yassine, 2004). Gershenson et al (1999) develop product modularity measures which are applicable at any life cycle of the product. In a similar vein, Newcomb et al (1998) discuss modularity at a module level of the product. However previous research has not focused on the measurement and usage of modularity at a component level, to develop operational and strategic plans (see the literature review section for a more detailed review of the modularity measurement literature).

We believe that component level modularity measurement is extremely important. On one hand, component modularity measures provide design and sourcing engineers with a basic tool for assisting in the day-to-day operational work, while on the other hand, these measures provide supply chain managers with a base-level tool that they can use to develop long term recommendations for sourcing.

Thus, our proposed network representation of products, focusing on component level modularity, can augment the way architectural properties of product components are defined in the literature and the way in which operational and strategic decisions are taken by practitioners.

This paper is structured as follows. First, the relevant literature in the product architecture and sourcing domain is reviewed. Next, some basic tenets of graph theory are presented and results are used to develop network representation of products leading to definition and development of component level modularity measures. This section is the heart of the paper. In the next section, these definitions are applied to determine the modularity of the components of an automobile subsystem (dataset described as in Pimmler and Eppinger (1994)) and a Delta Jigsaw (dataset available at the Design Repository of the Design Engineering Lab at University of Missouri-Rolla). Next, the Modularity Sourcing Framework is developed, taking component obsolescence as a policy decision variable. Effects of

modularity on sourcing for the two products are discussed. The paper concludes with a discussion of the results and comments for future work.

2. Literature Review

This section builds upon two streams of research. The first is the body of work dedicated to product architecture representations, and the second is the established stream of work focused on sourcing.

2.1 Product Architecture and Modularity

The literature on product decomposition and product architecture goes a long way back. Simon (1981, first edition of the book was in 1969) suggested that a product is a complex system, which is made up of many interacting parts. Each part is subordinated to the product system hierarchically. To simplify the complexity of the system, the product should be designed as a set of sub-assemblies (sub-systems) so that their assembly constitutes a new product. Through product modularization, the manufacturer can create many products by assembling different sub-assemblies within a short product development lead time. Alexander (1965) described the design process as a breakdown of designs into smaller subsystems that are minimally or loosely coupled. When the product sub-systems are significantly independent, the product redesign is limited to the modification of a set of related sub-systems, which could be done independently. This helps in improving the agility of the change management processes and overall reduction in design cycle time. Suh (1990;2001) builds upon these concepts by modeling the functional requirements of product design in terms of exchanges of energy, materials, and signals between functional elements organized in hierarchical function structures. Products have been considered as graphs of connected components and component connectivity is a central concept when studying engineering changes and design propagation during the development of complex products (Clarkson et al, 2004).

A key feature of product architecture is the level to which it is modular or integral. In the engineering design field, a large stream of research has focused on methods and rules to map functional models to physical components (Kirschman and Fedel, 1998; Newman, 2001; Suh, 2001; Eckert et al, 2004; Jaratt et al, 2004). However, as Ulrich and Eppinger (2004) have suggested, product architecture study involves mapping of functional elements to physical components as well as the specification of the interfaces among interacting components.

The Design structure matrix (DSM) is the basic tool for studying the structure of product architectures in terms of subsystem and component interactions. The DSM is a graphical method introduced by Steward (1981) and used by many researchers (see Eppinger et al, 1994 for example) to study interdependence between product development activities. Pimmler and Eppinger (1994) also used the DSM to illustrate product design decompositions. They posited that by using the DSM, development teams can better understand the complex interactions within the product system, thus simplifying the development process for large and complex projects. The DSM representations of complex products have also been extended to analyze the model design change propagation (Clarkson et al, 2004; Eckert et al, 2004; Jaratt et al, 2004). These papers specifically focus on the modes by which component level interactions impact the effect of any design change in any component.

Modularity is usually defined in the literature as an efficient way of organizing complex products and processes , by decomposing complex tasks into simpler portions so they can be managed independently and yet operate together as a whole (Baldwin and Clark, 2000). When designing complex products, modularity is considered an important product characteristic that results from directly mapping the functional and physical components of the product (Ulrich and Eppinger, 2004). From a systems perspective, modularity can be viewed as a continuum describing the degree to which a system's components can be separated and recombined, and it refers both to the tightness of coupling between components and the degree to which the "rules" of the system architecture enable (or prohibit) the mixing-and-matching of components (Schilling, 2000). Modularity permits components to be produced separately, or 'loosely coupled' (Orton and Weick, 1990; Sanchez and Mahoney, 1996), and used interchangeably in different configurations with very little effect on the overall system level performance or quality (Garud and Kumaraswamy, 1993).

How does one measure modularity? Ulrich and Eppinger (2003) propose that "Modularity is a relative property of a product architecture. Products are rarely strictly modular or integral. Rather, we can say that they exhibit either more or less modularity than a comparative product. (Page 166)) .They also propose three types of modular architecture :

(a) Slot-modular - where each of the interfaces between the major building blocks (called chunks) of the product are of a different type - so that various blocks cannot interact for example an automobile radio.

(b) Bus -Modular - Here there is a common bus to which other chunks connect via the same type of interface for example in an expansion card for a PC .

(c) Sectional-modular - where all interfaces are of the same type, but there is no single element to which all other chunks attach - example would be office partitions.

There have been various measures developed for describing the product architecture and its modularity. Gershenson et al (2004) provide a recent review of the measures developed in the literature for modularity. While mostly the measures developed resemble the DSM concept, there are two studies which offer new interpretations for modularity measurement. Newcomb et al (1996) use a multiplicative measure of modularity. The inter module connections are multiplied with the average correspondence between modules to arrive at a single number. This measure precisely links the material compatibility issues related to modules. Gershenson et al (1999) proposed an additive measure of modularity – they focused on developing a measure which can be applied during the complete product life cycle. Their measure consists of the addition of two ratios – the first ratio is the ratio of intra module similarities to those of all the similarities in the product (ie. both intra and inter module similarities). The second ratio is the ratio of of intra module dependencies to all the dependencies in the product.

A recent paper by Mikkola (2006) focuses on developing a new measure for the degree of modularization embedded in product architectures. This paper is in line with that of Mikkola and Gassman (2003) and develops a firm level view of modularity – The author takes four different factors that contribute to modularity: components (standard and new-to-the-firm), interfaces (standardization and specification), degree of coupling, and substitutability. A modularization function is developed to capture the effects of these four factor – the paper also describes how this measure can be used to elicit the opportunities for modularization of products.

2.2 Modularity and Sourcing

Aligning the decisions on modular product design and sourcing as well as overall supply chain design and coordination can not only save production costs (Ernst and Kamrad, 2000), but also improve supply chain performance (Fine, 1998). There are many papers focused on the integration of product modularization and supply chain design and coordination to optimize both operational and supply chain performance (Krishnan and Ulrich, 2001). This review is focused on the literature that integrates the concepts of design capabilities and design details like modularity with sourcing policies and supplier development routines.

Novak and Eppinger (2001) focus on how product architecture (specifically modularity) of components affects sourcing decisions. They use an original dataset and an interesting methodology. For simultaneously determining Product Complexity and Vertical Integration factors, they treat these two variables as jointly endogenous. Their main results show that complex products (say an engine) have interfaces that need coordination for development of design. It may require much more time to coordinate the sourcing with suppliers outside the firm than within. Therefore, in-house development of complex products requires less effort in coordination and redesign. Their results also show that

i. As product complexity increases, firms tend to vertically integrate

ii. Product quality can be ensured when manufacturers design simpler products for outsourcing to module suppliers.

iii. Product design and supply chain design both need to be harmonized for superior performance. This indicates that product design engineers and supply chain executives need to work in close coordination within the firm to effect optimal decision making.

Sanchez and Mahoney (1996) posit that an increase in modularity leads to more outsourcing. They reason that this effect is induced because the standardized component interfaces in a modular product architecture reduce the coordination cost of trading at arm's length. Schilling (2000) links modularity to industry standards. She argues that industry-wide standardization — de facto as well as regulatory — makes the interrelation between components very generic, which leads to an increase in modularity and incentivizes outsourcing policies.

Ulrich and Ellison (2005) focus on the motives for internalizing an activity within a firm and posit that decisions about internalizing design and internalizing production cannot be fully understood in isolation. They conclude that design and production activities can only be disintegrated when production processes have matured to the point where there are explicit design rules that express the constraints and capabilities of the production process. Fine (1998) suggests that module suppliers have greater autonomy and need lower proximity to improve supply chain performance. As each product module is independent of the others, the supplier is only required to conform to the predefined module specifications, but not to consider the modifications of other modules. Schilling (2000) notes that modularized components allow suppliers to work on particular modules by themselves and still assure that the modules will interact effectively in the product development process. Therefore the problem of iterative communication and coordination between suppliers and manufacturers in product development modification and engineering change management is reduced (Ulrich and Eppinger, 2004). Fine (1998) also suggests that a modular supply chain can result from developing

modular products. This may have effects on the physical location of the suppliers from the core assembly plants easing transaction costs and permitting interchangeable arrangements for major components. The effect of collocation was also focused on by Dyer (1996) in an interesting paper on the differences between asset specialization of US and Japanese firms. He finds that firms which have tightly integrated production network have better performance. An interesting part of Dyer's paper looks at the configuration of Japanese automotive plants and their suppliers. He documents that all Toyota's plants are within 32 kms of each other and affiliated supplier plants are 30.7 miles away (on average) with independent supplier's average distance being 86.6 miles. Nissan's suppliers are 53 kms (affiliated) and 172 miles (independent) away from its main plants. For General Motors (GM), the plants are scattered around US and primary GM suppliers are 350 miles away on an average. He posits that geographic proximity is one of the reasons of higher asset specificity of Toyota. The advantage of close supply chain design with manufacturers and suppliers (e.g. physical collocation) improves the chances of face-to-face communication and joint product development between them, leading to better tacit knowledge sharing which is vital for product innovation.

The concept of better knowledge sharing leading to better sourcing, and its link to component modularity has been explored in a paper by Gerwin (2004) who posits that, in the contractual relationship between buyers and suppliers, coordination requirement (referring to the total intensity of information processing needed in product development) and the ability of coordination (defined by the number of available coordination methods) in modular product development are lower than that in integrated product development. Similarly, In their case study of product modularization on supply chain design and coordination in Hong Kong and China, Lau and Yam (2005) show that product modularization reduces product development time, and improves product quality and inventory levels. They further posit that supply chain design is greatly affected by product modularization while supply chain coordination is affected by whether the product is innovative or conventional.

Overall, the current literature links modularity to sourcing decisions and we can conclude that modular products are better candidates for outsourcing. This literature takes as given the concept of 'core competencies', first forwarded by Prahlad and Hamel (1994). At a strategic level, a firm can focus on its knitting and outsource the operations which are not in its core domain. The modularity and sourcing literature focuses on the detail of operationalizing the 'non-core' operations. But are there other variables which affect this conclusion? How can managers operationalize the impact of other variables that moderate the decision of outsourcing? In section 5, We look at a moderating variable of obsolescence for enriching this discussion.

3. Graph Theory and Network Representation of Products

In this section, a product network representation based upon graph theory is developed using the foundations to define properties of products when considered as graphs of connected nodes.

3.1 Graph Theory

For a detailed discussion on graph theory and its application to network concepts, the reader is referred to Harary (1994) and Diestel (2005). Some basic graph theoretic concepts are presented below which will help build the connections to product networks literature.

A graph is a symbolic representation of a network and of its connectivity. The fundamental mathematical entity is the binary directed graph or digraph. A digraph is a set of nodes and a set of links which connect pairs of nodes. The basic units of analysis under our approach consist of these nodes and links. Nodes are the junctions that represent the critical points of origin, routing, and termination. Links are any type of connection between nodes. The adjective binary represents the added constraint that we do not allow the links to have strengths or that the links may be of different types. Thus the digraph is a mathematical representation of the simplest form of choice data – unranked choices on a single criterion.

A basic network configuration can be one that directly links every pair of nodes. Such a network is called a complete graph (Figure 1). If a network consists of n nodes, then its complete graph will have (1/2) n (n – 1) links. A complete graph also corresponds to a point-to-point network. A prominent feature of point-to-point networks is that they contain numerous cycles, which are paths along which it is possible to pass through a succession of links and eventually return to the original node without crossing any link more than once. A cycle is thus a closed path, with no other repeated nodes than the starting and ending nodes. (This is often related to what is known as the first problem of graph theory. Euler studied whether it was possible to cross each of the seven bridges interconnecting the two banks of the Pregel River and the island of Kneiphof, located within the city of Königsberg, without crossing any bridge more than once. Euler posited that every node except for the beginning and ending nodes of the path must necessarily have an even number of links leading away from it if the type of path that Euler sought were to exist. Because the network created by the Königsberg bridges contained four nodes with an odd number of links, no such path existed. This description is in Barabási (2002), note on page 12).

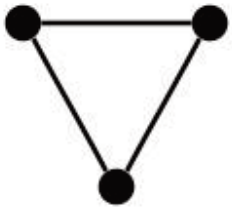

Figure 1. A Connected Graph

The network architecture that minimizes the number of links is a tree, which is a graph that connects nodes without creating any cycles. A tree that connects all of the nodes in a network is known as a spanning tree. In a network with n nodes, such a spanning tree would consist of n – 1 links. According to Cayley's Formula (Harary, 1994), the number of spanning trees in a graph with n nodes is nn–2.

Another fundamental concept in graph theory is the geodesic, or the shortest path of nodes and links that connect two given nodes. There may not be a unique geodesic between two nodes: there may be two or more shortest paths, which may or may not share some nodes. An Algorithm to calculate geodesics has been given by Newman (2001).

3.1.1 Network Adjacency Matrix, X

Let us denote the number of nodes in a graph as g – the size of the group. If the nodes are numbered arbitrarily from 1 to g, then we may construct a useful matrix representation of the

$$X_{ij} = \begin{cases} 1, & i \rightarrow j \\ 0 \text{ Otherwise} \end{cases}$$

digraph as below. Let X be a g*g matrix whose (i,j) entry is

Note that i -> j means that there is a directed link from node i to node j. There may also be a directed line from node j to node i, but this possibility is neither implied nor denied by the notation . The link (i , j) is of initial extremity i and of terminal extremity j. The matrix X is called the *Adjacency Matrix* in graph theory.

A weighted graph is an extension of digraph where we relax the assumption of the links having no strengths. A weighted graph has a number associated with each link. The numbers are called link weights and the graph is said to have weighted links. Like links, nodes may also be weighted. A graph in which every node is associated with one or more numbers, called node weights, is referred to as a graph with weighted nodes. Link weights are often used to represent some physical parameter of interest in applications of graph theory. For example, a graph may represent a system of roads between five nodes A, B, C, D, and E, where the numbers attached to each link, the link weights, can represent the length in kilometers of that link.

A *network* is a graph with particular numerical values, such as cost or capacity or strength of relationship between the nodes, assigned to the links. Thus, a network is an extension of the digraph and the matrix X associated with a network is called the Network Adjacency Matrix. The architecture of a network refers to the set of nodes and the pattern of the links that connects them.

3.1.2 Degree and Bridge of X

In graph theory the degree or valency of a node i is the number of links incident to i. Let deg(i) denote the degree of i. The variable deg(i) therefore ranges from a minimum of 0 to a maximum of (n-1) if there are n nodes in a graph.

In a directed graph the indegree of a node v is the number of edges terminating at i and the outdegree is the number of edges originating at i. Let deg + (i) and deg − (i) denote the indegree and outdegree of node i. The degree of a node deg (i) is the sum of its deg+(i) and deg- (i). Note that the sum of deg+(i) over all nodes equals the sum of deg -(i) (and both are equal to n for a digraph). Individual nodes may show imbalances in their indegree and outdegree.

A node with deg(i) = 0 is called isolated. A node with deg(i) = 1 is called a leaf. If each node of the graph has the same degree k the graph is called a k-regular graph and the graph itself is said to have degree k. A node with deg + (i) = 0 is called a source and a node with deg − (i) = 0 is called a sink.

For a weighted graph or a Network, the calculation of indegree and outdegree is moderated by the weights of the links. Our variable Xij then is no longer binary but can have a value greater than 1 also. The indegree will then be defined as

$$Deg+(i)=\frac{\sum_{j=1}^{n} X_{ji}}{X_{\max}} \quad \ldots\ldots(1)$$

where X_{max} is the maximum strength of any of the links and therefore the maximum value of X_{ij}. We divide by X_{max} to make sure the degree measure is homogeneous across all nodes of the graph.

The outdegree is defined similarly as

$$Deg-(i)=\frac{\sum_{j=1}^{n} X_{ij}}{X_{\max}} \quad \ldots\ldots(2)$$

where the only change on the right hand side is that the subscripts of X have changed to indicate the outgoing direction of the links at node i.

A useful definition in graph theory for developing connections to the product architecture is that of a *bridge*. The bridges of a connected graph are the graph links whose removal disconnects the graph . Harary states ' A bridge is an edge of a graph G whose removal increases the number of components of the graph G' (1994, p. 26). (An edge is a link, I use the term link throughout to maintain homogeneity). We can note that every link of a tree is a bridge. Figure 2 shows an example of a graph with five nodes. The links which are not at the end are bridges.

Figure 2. The three middle nodes are bridges

As we had noted before, a geodesic is the shortest path of nodes and links that connect two given nodes. If we calculate the ratio of all geodesics between two nodes, a and b, which contain our focal node i ($nd_{ab}(i)$) to the number of total geodesics between a and b (nd_{ab}) we will get a measure of how much bridging is being done by node i – that is we will get a measure of how much 'in between' a and b the node i is. Here nd is not the geodesic distance d but the total number of these geodesics between a and b. Summing over all such pairs of a and b components in the product give us a measure of the bridging strength of node i.

$$Bridge(i)=\sum_{a<b, i\neq a, i\neq b} nd_{ab}(i)/nd_{ab} \quad \ldots\ldots(3)$$

3.2 Product Architecture and the Design Structure Matrix

We will use the standard tool of product architecture, the DSM or the design structure matrix, (Eppinger et al, 1994; Sharman and Yassine, 2004) for developing the link to product architecture literature. Let us denote the DSM by **Y**. **Y** is a square matrix whose columns and rows are identically labeled with the components of the product. It is the matrix of design dependencies for any type of design dependency - Previous work in engineering design has identified various types of design dependencies between components such as spatial, structural, material, energy, and information (Pimmler and Eppinger, 1994). Hence, **Y** captures the dependency between components for any given design domain. Y has non-zero elements, Y_{ij}, if component *i* depends for functionality on component *j*. The value of Y_{ij} indicates the strength of the design dependency.

We immediately see the parallel between the Network Adjacency Matrix defined in graph theory and the Design Structure Matrix defined in Product Architecture literature. *Matrices **X** and **Y** are similar in their representation : **X** is a group of nodes and links and **Y** is a group of components and their dependencies. These two representations – from two different research streams - are similar in their nature*. It is proposed that graph theoretic formulations can aid the DSM formulations by helping the practitioners via the network representation of complex products. Such a representation helps to identify modular components which can aid policy decision on design and sourcing.

3.3 DSM and Network Representations – An illustrative view

To further illustrate the parallel between the Network representations and the DSM representations, we take an example of a product called the Delta Jigsaw. Table 1 shows the DSM of the Delta Jigsaw. This is a 41*41 matrix and represents the design dependencies of all the 41 components of the Delta Jigsaw with each other. Treating this matrix as a Network Adjacency Matrix, we can identify each component as a node and the design dependencies as links of a graph. The network representation of such a graph is shown in figure 4 (drawn with UCINET). This network representation helps us in identifying the modular and integral parts of the Network (at a component level) by detailing the interconnectivity of the network. For example, figure 4 details that a component 'switch' has a lot of direct design dependencies with other components and thus represents a component that is very embedded in the overall design of Delta Jigsaw. If there are changes in the design of switch, there may be associated design changes in other linked components. We can then think of the *switch as a component that is very integral to the design of a Delta Jigsaw or alternatively has very low modularity for the Delta Jigsaw.* Such a representation then gives us inspiration to define modularity of the components in terms of their dependencies on other components. There may be other ways in which components could affect each other and these different ways help us in developing different measures of component modularity.

Such representations have gained popularity in understanding the product architecture.

#	COMPONENT	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
1	battery	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	battery contact 1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	battery contact 2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	battery plug	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	blackwire	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
6	blade	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
7	blue wire	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
8	cam	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	debris shield	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
10	drivetrain	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	electrical	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	gear arm	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	guide mount	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	guide plate	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	handle	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	jigsaw	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	left casing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	motor	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	motor mount	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	output	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
21	pin	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	red wire	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
23	right casing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	saftey button	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
25	saw blade door	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	saw guide	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	saw lock	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	screw 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
29	screw 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
30	screw 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
31	screw 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
32	screw 5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
33	shell	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0
34	solder	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
35	spring	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
36	switch	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	1
37	system	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0
38	washer 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
39	washer 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
40	washer 6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
41	white wire	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1

Table 1 : Design Structure Matrix of the Delta Jigsaw (Equivalent to Network Adjacency Matrix) .

What does this matrix represent? The 41 41 matrix is a matrix of 0's and 1's.*

The first column has the names of the 41 components that make up the product called "Delta Jigsaw". The first row also has these 41 components, however for representation the product names have been replaced by their component numbers (1 to 41). If there is 1 in a particular cell, this means that the row component and the column component have a dependency with each other.

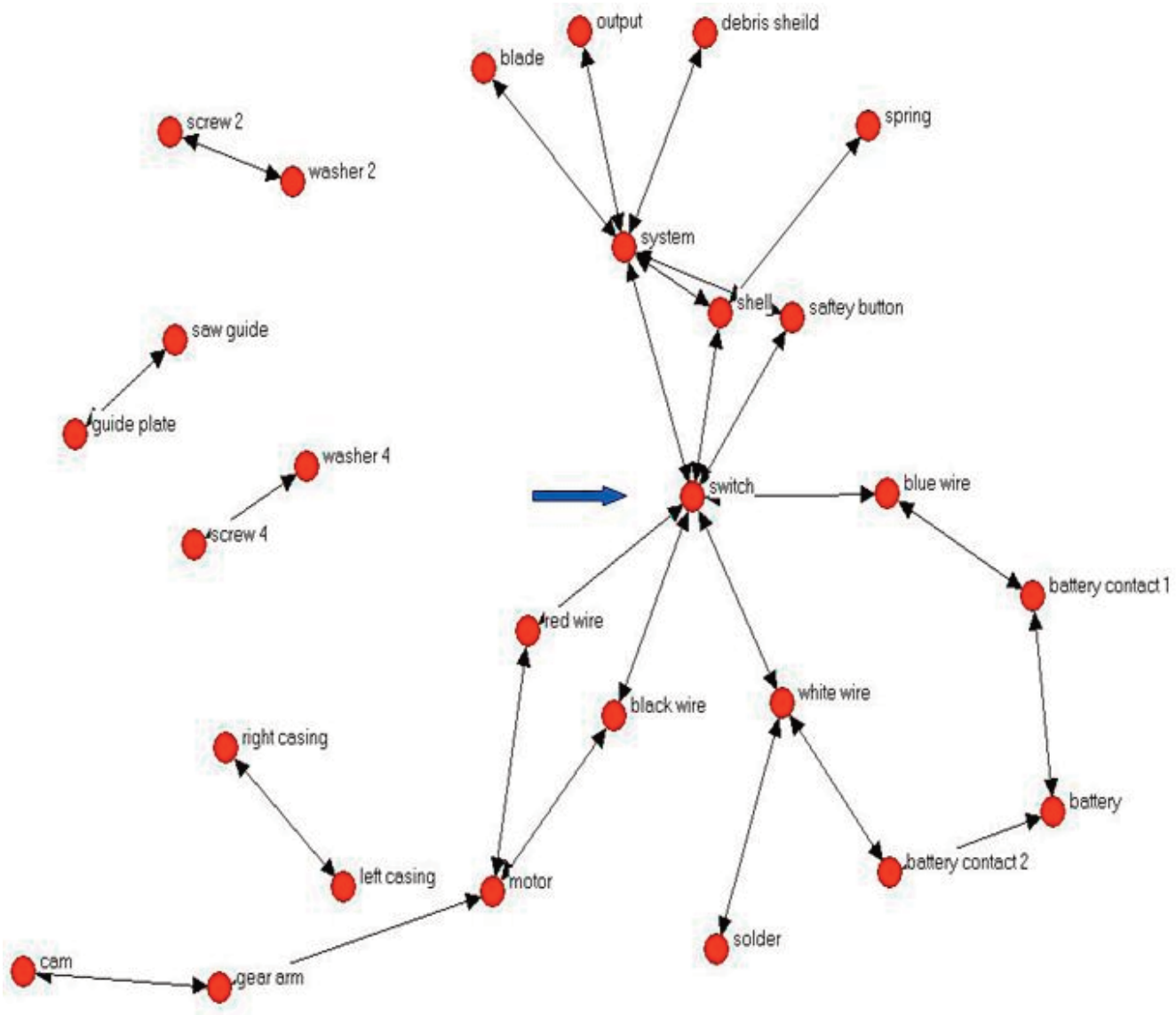

Figure 4 : Product Network Representation of the Delta Jigsaw.

What does this representation show ? In essence, there is the same information as that in matrix of figure 1. However, we see that this network representation gives us a much better appreciation of the links between components. As discussed in the text, the component called switch (see arrow) seems to have a lot of dependencies with other components, indicating that it may not be modular.

(For clarity, components which do not have links to other components are not shown – they are isolates in graph theory parlance)

4. Component Modularity Definitions

In this section, we develop formal measures of component modularity based on the discussions of graph theory and the similarity of representations of the Network Adjacency Matrix and the Design Structure Matrix, which we developed in section 3. We link the proposed measures to graph theory and product architecture literature.

4.1 Degree Modularity

The most fundamental measure of modularity that we propose is the answer to the question: "how many other components depend on the design of this component?" The larger the number of components that affect, or are affected by, the design of component *i*, the less modular component *i* is. It is clear then, that the modularity of a component can be defined precisely as an inverse of the definition of degree of a node as detailed above from graph theoretic considerations.

Similar to graph theory definitions given above, the *In-Degree* of a component *i* is equal to the number of other components that *i* depends on for functionality, whereas *Out-Degree* is equal to the number of other components that depend on component *i*. Thus we define, for a product with n components, the *In-Degree Modularity* of component *i, M(ID)i*, as

$$M(ID)_i = \frac{Y_{\max}}{\sum_{j \neq i, j=1}^{n} Y_{ji}} \quad \ldots\ldots(4)$$

where Y_{max} is the maximum value that Y_{ji} can take.

Similarly, the *Out-Degree Modularity* of component *i*, *M(OD)i,* can be defined as

$$M(OD)_i = \frac{Y_{\max}}{\sum_{j \neq i, j=1}^{n} Y_{ij}} \quad \ldots\ldots(5)$$

where again, Y_{max} is the maximum value that Y_{ij} can take. Thus the degree modularity of a component will be the sum of the above and will be given by

$$M(D)_i = \frac{Y_{\max}}{\sum_{j \neq i, j=1}^{n} Y_{ij}} + \frac{Y_{\max}}{\sum_{j \neq i, j=1}^{n} Y_{ji}} \quad \ldots\ldots(6)$$

We see that this definition is probably inadequate for product network representation from the view of a firm, since the absolute values that the above definitions will give will not be interpretable across products. Hence we need to standardize the above definitions so that degree modularity of a component is an interpretable value across products. One way to do this would be to introduce the number of components we want to measure (n) in the definition. We can also constrain the measures so that the value obtained is between 0 and 1. Applying these modifications, we get the standardized measures of indegree modularity, outdegree modularity and the degree modularity of a component. The standardized measures then become

$$M_{std}(ID)_i = 1 - \frac{\sum_{j \neq i, j=1}^{n} Y_{ji}}{n.Y_{max}} \qquad (7)$$

$$M_{std}(OD)_i = 1 - \frac{\sum_{j \neq i, j=1}^{n} Y_{ij}}{n.Y_{max}} \qquad (8)$$

so that the standardized degree modularity measure for a component is

$$M_{std}(D)_i = 2 - \frac{\sum_{j \neq i, j=1}^{n} Y_{ij} + \sum_{j \neq i, j=1}^{n} Y_{ji}}{n.Y_{max}} \qquad (9)$$

A high value of any of the modularity measures indicates that there are fewer and/or weaker design dependencies and therefore the component is more modular. The maximum value of degree modularity is 2, which corresponds to a component that has zero design dependencies with all other (n-1) components of the product. Hence, such a component would be highly modular. In graph theory terms, this component is an isolate node that has no links to any other node in the graph.

4.2 Bridge Modularity

A second way of measuring modularity is akin to the definition of a bridge in graph theory. Here we can focus on those components that link two highly integral components. We can view these *bridge* components as having control over the design dependency flow since information about the design dependency must propagate through them. In this sense, these components can be considered as information valves that regulate the amount of information transmitted in the product network for some dependencies. The more a component is "linked to" other integral components, the more integral it is – thus a less modular component is less related to integral components. What is meant is that even if a component has only two links, if the components at the other ends of these two links have a very low modularity, the modularity of our focal component should also be low.

As noted in the previous section, graph theory definition of a bridge is a line such that the graph containing the line has fewer components than the subgraph that is obtained after the line is removed. In product representation domain, we can then think of components becoming more integral as their bridging position increases. As a result we define *bridge modularity* of component *i* based on the number of times it is on the path of two other components. We can assume that components lying on most geodesics will be the one bridging most components and therefore the least modular. This assumption makes sense in the product domain if a design dependency between two components propagates through the minimum number of parts (i.e. the geodesic). Thus the bridge modularity of a component can be defined precisely as an inverse of the definition of bridges of a node as detailed from graph theoretic considerations. Hence, if we calculate the ratio of all geodesics between two components, *a* and *b*, which contain our focal component *i* ($nd_{ab}(i)$) to the number of total geodesics between *a* and *b* (nd_{ab}) we will get a measure of how "in the middle" (between a and b) component *i* is. Summing over all such pairs of *a* and *b* components in the product give us a measure of the bridging potential of component *i*. Our measure *M(B)* then takes the form

$$M(B)_i = \frac{1}{\sum_{a<b, i \neq a, i \neq b} nd_{ab}(i) / nd_{ab}} \qquad (10)$$

Once again, we see that this measure will give us different values for different products depending on number of components. We can standardize this measure by taking into account all pairs of components excluding component *i*. There can be (*n*-1) components not including *i,* which can have geodesics with (*n*-2) other components. Note that the fewer geodesics component *i* is on, the higher the value of $M(B)_i$, and the more modular component *i* is. To have measure which is similar to our degree modularity standard measure, we constrain our bridge measure to lie between 0 and 2. Our standard measure of bridge modularity then becomes

$$M_{std}(B)_i = 2 - \frac{(n-1)(n-2)}{\sum_{a<b, i\neq a, i\neq b} nd_{ab}(i)/nd_{ab}} \quad \ldots\ldots(11)$$

Similar to the degree modularity measure, the maximum value of this index is 2, which is reached for a perfectly modular component that is not on the geodesic of any other pair of components – then our focal component *i* does not bridge any two other components in the product for that particular type of design dependency. An Algorithm to calculate geodesics is given by Newman (16) and is also available in commercial software packages like Mathematica.

Bridge modularity measure is quite different from the degree modularity measure in its focus of measurement. Since bridges are nodes that disconnect the network if removed, a low bridge modularity measure would mean that the particular component is one which is critical – if such a component malfunctions, the product network will disconnect – the product cannot function. Thus, while degree modularity measures the direct effect of design dependencies, bridge modularity measures the sensitivity of these dependencies as they are propagating through the product network. Bridge modularity is thus a representative measure of the critical and sensitive areas in the product network.

I consider the two proposed measures of component modularity to be complementary of each other because they emphasize related but distinct features about the patterns of design interfaces between product components. *Degree modularity* only takes into account the effects of immediate neighbors neglecting the connections beyond adjacent components. In addition, it captures the strength of the design dependency. Since the design structure matrix need not be symmetric, we define *In-Degree* and *Out-Degree* modularity. Bridge modularity is based on the component's role in bridging other components and therefore its sensitivity with regards to the overall functioning of the product. Thus bridge modularity does take into account the effect of components which may not be its immediate neighbors. The less bridging role a component has, the more modular it is.

Both these measures are based on the underlying argument that the more independent the components are from other components, the more modular they become. Less modular components are components with many interfaces and/or occupying bridging positions in the product. We therefore formally define component modularity. Component modularity is defined as the *level of independence of a component from shared interfaces in a product*. This definition implies a range of modularity at the component level. This definition also entails that constraints on components due to their interactions with other components define their modularity.

In the next sections, this understanding of component modularity and the modularity formulations are applied to two different products for which design structure matrices have been developed. We will then explore how we can build on the modularity knowledge of the components to evolve sourcing decisions.

5. Sourcing Policy Development

In this section, I develop a theoretical policy decision framework linking sourcing of components to component level modularity. While one can focus on a number of parameters for sourcing, I focus on a single parameter - the obsolescence of the product. In today's automotives, which are having large interfaces with electronics, there is an increasing degree of obsolescence built in. Some components may have a longer life cycle (example castings and forgings) while others may have a shorter life cycle (example electronic chips – faster chips may come in very soon). In an interesting essay, Saleh (38) gives the example of the obsolescence of the flight management system of the Boeing B-777 airplane. The Boeing 777 relies on the Intel 80486 chip for its Flight Management System. The airplane was designed to be in use for approximately 30 years. The problem is that Intel will withdraw support to 486 chips by next year. Saleh notes "The Flight Management system must be (or should have been) designed to accommodate flexibility (Saleh, 2005)." The upgrade costs have been estimated to be close to $250,000 per circuit redesign.

Component obsolescence can be mitigated in many ways :

- By arranging for alternate or substitute parts
- By Cannibalizing from product returns
- By procuring from Grey Market or After market
- By having a Lifetime buy – Firms can stock the lesser lifetime part for the life of the product/system !
- By Reverse engineering or Process emulation
- But in a proactive way – by Sourcing decisions based on modular and integrative component detailing.

In this section we explore the last alternative - How can component level modularity be applied to design/sourcing decisions so as to obviate the risks of obsolescence? The guidelines we develop can be used for improving component level sourcing at a firm level. At the sourcing level, the steps related to this decision may be the following

1. The network of components for the complex products can be explored using the DSM and the product representations to measure component level modularity. The degree and the bridge modularity values of all components can be then computed.

2. Component obsolescence can be measured by evaluating industry trends and product life cycle estimates.

3. Finally, the procurement policy for each component can be developed to evolve a coherent sourcing policy.

We develop below the sourcing policy framework on the twin parameters of obsolescence and component modularity. For modular components that also have a high obsolescence profile, the policy decisions can focus on outsourcing. Since the product clockspeed (Fine, 1998) is fast, firms who are OEM's (and their product obsolescence rate is lower than the focal component) may ideally not invest in the technology required for upgrading the components. Thus, a highly modular component with a high obsolescence rate is a candidate for the "buy" process within the make-buy sourcing decision process. On the other hand, a component that has a very low modularity (and is therefore integral to the product) but also has a high obsolescence rate is a candidate for maintaining technological edge. This can be done either by developing very strong supplier relationships, or by developing the component in-house – in either case, the firm has to continue investing in competencies to ensure that the design dependencies that the focal component has with other components does not affect the product performance, service levels and warranty commitments, even after the product has been introduced.

For the products which have a high modularity but a low obsolescence rate, the decision is not so complex. Such components are ideally purchased from the market as per the needs. For the last remaining combination of a component with low modularity and low obsolescence, the firms will normally develop these components in-house or have strong processes built around these components by virtue of previous associations. The above discussion is summarized on the two axes of modularity and obsolescence in figure 5.

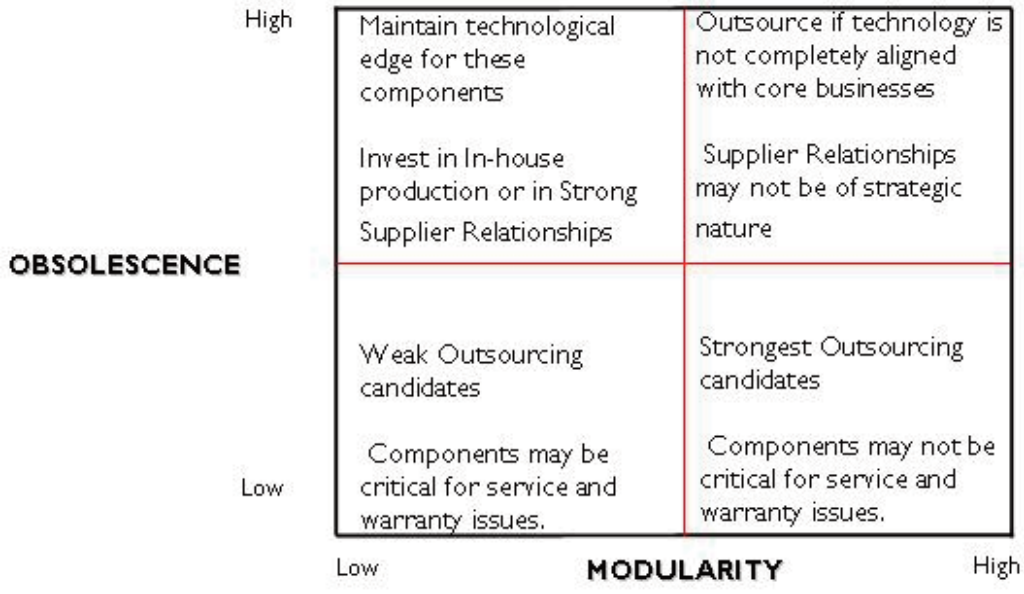

Figure 5 – The Modularity Sourcing Framework

So, can the above discussion lead a manager to a simplistic argument – "If a component is of low modularity then we should develop the component in house, and if it is of high modularity - buy it off the shelf? – I am not too sure how obsolescence comes in? " [1]

The reasoning is slightly more involved. We propose that the obsolescence variable is a moderator between sourcing and modularity. A highly integral component may be a good candidate to produce inhouse - however, obsolescence considerations can dictate that the product designers go back to the drawing board and redesign the components so that a modular architecture results. Such architecture will then dictate outsourcing - which is a different decision than that taken in absence of figure 5. This is what we mean by a moderator variable - obsolescence affects the strength of the relationship between sourcing and modularity. We can also understand the moderating relationship as an interaction. The relationship between the sourcing and modularity variables depends on the level of obsolescence.

How is this discussion of obsolescence linked to component modularity? Having a value of modularity at the component level allows the designers to experiment on components and modules *that have the most potential for altering the value of the system*. Performing many experiments on the components most critical to overall system performance maximizes the overall value. Because of the computation of modularity at the component level, the designer now has the option to pick the best outcome from many trials.

6. Applications

6.1 Applications I - Delta Jigsaw

We use the dataset available at the Design Repository of the Design Engineering Lab at University of Missouri-Rolla. The DSM is shown in figure 3 and the Network Representation in Figure 4. As discussed earlier, the network representation lets us visually analyze the interconnections between the components. The component modularity definitions developed in the previous sections are used to confirm the intuition developed via the product network representations. Both the modularity measures are calculated for all the components of the Delta Jigsaw. Table 2 shows the results of the modularity calculations. The most modular components can be identified easily using the modularity values. Since we had standardized our measures, these values are also comparable with other products to get a sense of the level of modularity of components within a product.

Analysis of Delta Jigsaw Modularity Values

The data values are aligned with the primary intuition developed via the network representation. The basic measure of degree modularity is what corresponds to the network graphs, and we see that two components - switch and system - show high integral values. The bridge modularity values provide added information, needed for day-to-day operations and policy level decisions. We recollect that a bridge represents a node that. if removed, will disconnect the network. Thus, the bridge modularity values provide us with the 'sensitive' components – those whose operation is critical to the functioning of the network as a whole, even though they may not be highly integral from a degree perspective.

We see that the battery and its contacts have the lowest values of bridge modularity. These then, are the sensitive spots for operation of the Jigsaw as a product. If one of these components fail, the Jigsaw fails. This provides the designers with added information about the needed performance parameters.

Overall, we can interpret that components with low degree modularity values provide direct information about the modularity or integrality of that component while the low bridge modularity values provide information about the sensitive spots in the product. These results can then be applied on the Modularity Sourcing Matrix (Figure 5) to develop sourcing recommendations. For example, we can recommend that the component switch is a component whose design dependencies are the strongest. It is the most integral component of the Delta Jigsaw. If the product life cycle of the switch is such that its obsolescence is high, then the firm producing the Delta Jigsaw may like to either produce the switch in-house or develop strong technical competencies around the switch production. On the other hand, there needs to be enhanced quality control and design processes developed for the battery, the contacts and the wires to ensure that their design and production (either in-house or outsourced) is robust. There may be inventory recommendations developed for both in-house production as well as service related issues. For example, service centers may be advised to treat the switch, the system, the battery and battery contacts as critical items, items which must never be allowed to be out of stock, so that customer service satisfaction is high.

1 We are grateful to one of the referees for pointing this out.

#	Components	Degree Modularity	Bridge modularity
1	battery	1.90	**1.20**
2	battery contact 1	1.90	**1.75**
3	battery contact 2	1.90	1.79
4	battery plug	2.00	2.00
5	black wire	1.90	1.90
6	blade	1.95	2.00
7	blue wire	1.90	1.90
8	cam	1.95	2.00
9	debris shield	1.95	2.00
10	drivetrain	2.00	2.00
11	electrical	2.00	2.00
12	gear arm	1.90	1.88
13	guide mount	2.00	2.00
14	guide plate	1.95	2.00
15	handle	2.00	2.00
16	jigsaw	2.00	2.00
17	left casing	1.95	2.00
18	motor	1.85	1.94
19	motor mount	2.00	2.00
20	output	1.95	2.00
21	pin	2.00	2.00
22	red wire	1.90	1.90
23	right casing	1.95	2.00
24	saftey button	1.90	2.00
25	saw blade door	2.00	2.00
26	saw guide	1.95	2.00
27	saw lock	2.00	2.00
28	screw 1	2.00	2.00
29	screw 2	1.95	2.00
30	screw 3	2.00	2.00
31	screw 4	1.95	2.00
32	screw 5	2.00	2.00
33	shell	1.85	1.88
34	solder	1.95	2.00
35	spring	1.95	2.00
36	switch	**1.66**	1.98
37	system	**1.71**	1.96
38	washer 2	1.95	2.00
39	washer 4	1.95	2.00
40	washer 6	2.00	2.00
41	white wire	1.85	1.95

Table 2 : Degree and Bridge Modularity values for Delta Jigsaw

6.2 Applications 2 - The Climate Control System

We use the dataset published by Pimmler and Eppinger (1994) describing the climate control system of an automobile. The DSM is shown in Table 3. We divide the DSM into four different design dependency matrices to distinguish the four types of design dependencies between the physical components - Material, Spatial, Energy and Information. We note that this DSM uses a three-point scale to capture the level of criticality of each dependency for the overall functionality of the component in question – for all four dependencies. These metrics are discussed at length in Pimmler and Eppinger (1994).

There are therefore four design dependency matrices that result from the original DSM (Table 3). One of the resulting matrices – the Energy design dependency matrix is shown in Table 4 – it is basically a subset of table 3. The design dependency matrix in this case is similar to the network adjacency matrix and can be used instead of the DSM as the input for developing the modularity measures. I develop network representations for the four types of design interfaces between the components. Figure 6 shows the network representation for the material design interfaces while Figure 7 shows the same representation for the energy dependencies. These diagrams help us in visually analyzing the interconnections between the components.

The component modularity definitions developed in the section 4 are used to calculate modularity measures for all the components of the Climate Control System for all the four dependencies. Table 5 shows the results of the modularity calculations. The most modular and integral components can be identified easily using the modularity values, however the analysis is more complicated and therefore more insightful since we have four types of design dependencies identified. Since we had standardized our measures, these values are also comparable with other products to get a sense of the level of modularity of components within a product.

Climate Control System Interactions

Key: S E I M

		K	J	D	M	L	A	B	E	F	I	H	C	P	O	G	N
EATC Controls	K		0 0 2 0		1 0 0 0	0 0 2 0				0 0 2 0					0 0 2 0		0 0 2 0
Refrigeration Controls	J	0 0 2 0			1 0 0 0					0 0 2 0	1 0 0 0						
Heater Hoses	D										-1 0 0 0		1 0 0 0				
Command Distribution	M	1 0 0 0	1 0 0 0			1 0 0 0		1 0 0 0		1 0 0 0				1 0 0 0	1 0 0 0		1 0 0 0
Sensors	L	0 0 2 0			1 0 0 0												
Radiator	A							2 0 0 2	2 -2 0 0								
Engine Fan	B				1 0 0 0		2 0 0 2		2 0 0 2								
Condenser	E						2 -2 0 0	2 0 0 2		0 2 0 2		-2 2 0 2					
Compressor	F	0 0 2 0	0 0 2 0		1 0 0 0				0 2 0 2		1 0 0 2	0 2 0 2					
Accumulator	I		1 0 0 0	-1 0 0 0						1 0 0 2		1 0 0 2					
Evaporator Core	H								-2 2 0 2	0 2 0 2	1 0 0 2		-1 0 0 0	0 0 0 2		2 0 0 0	
Heater Core	C			1 0 0 0								-1 0 0 0		0 0 0 2		2 0 0 0	
Blower Motor	P				1 0 0 0							0 0 0 2	0 0 0 2		2 0 0 2	2 0 0 2	
Blower Controller	O	0 0 2 0			1 0 0 0									2 0 0 2		2 0 0 0	
Evaporator Case	G											2 0 0 0	2 0 0 0	2 0 0 2	2 0 0 0		2 0 0 0
Actuators	N	0 0 2 0			1 0 0 0											2 0 0 0	

Table 3 : DSM of the Climate Control System (from Pimmler and Eppinger ,1994)

*There are four design dependencies – Spatial , Energy, Information and Material. These are denoted by S,E,I and M respectively. Against each part, these dependencies are denoted in a 2*2 matrix. The key of these four dependencies is at the upper right corner of the table.*

This DSM table thus gives information about four separate dependencies. It can be broken up into four different matrices, each for one dependency. Table 4 gives an example of this breakup – the energy dependency matrix.

		K	J	D	M	L	A	B	E	F	I	H	C	P	O	G	N
EATC Controls	K		0		0	0				0					0		0
Refrigeration Controls	J	0			0					0	0						
Heater Hoses	D										0		0				
Command Distribution	M	0	0			0		0		0				0	0		0
Sensors	L	0			0												
Radiator	A							0	2								
Engine Fan	B				0		0		0								
Condenser	E						2	0		2		2					
Compressor	F	0	0		0				2		0	2					
Accumulator	I		0	0						0		0					
Evaporator Core	H								2	2	0		0	0		0	
Heater Core	C			0								0		0		0	
Blower Motor	P				0							0	0		0	0	
Blower Controller	O	0			0									0		0	
Evaporator Case	G											0	0	0	0		0
Actuators	N	0			0											0	

Table 4 : Design Dependency Matrix for ENERGY dependency for the Climate Control System .
(This table is a subset of Table 3. There are total 4 such matrices for four design dependencies.)

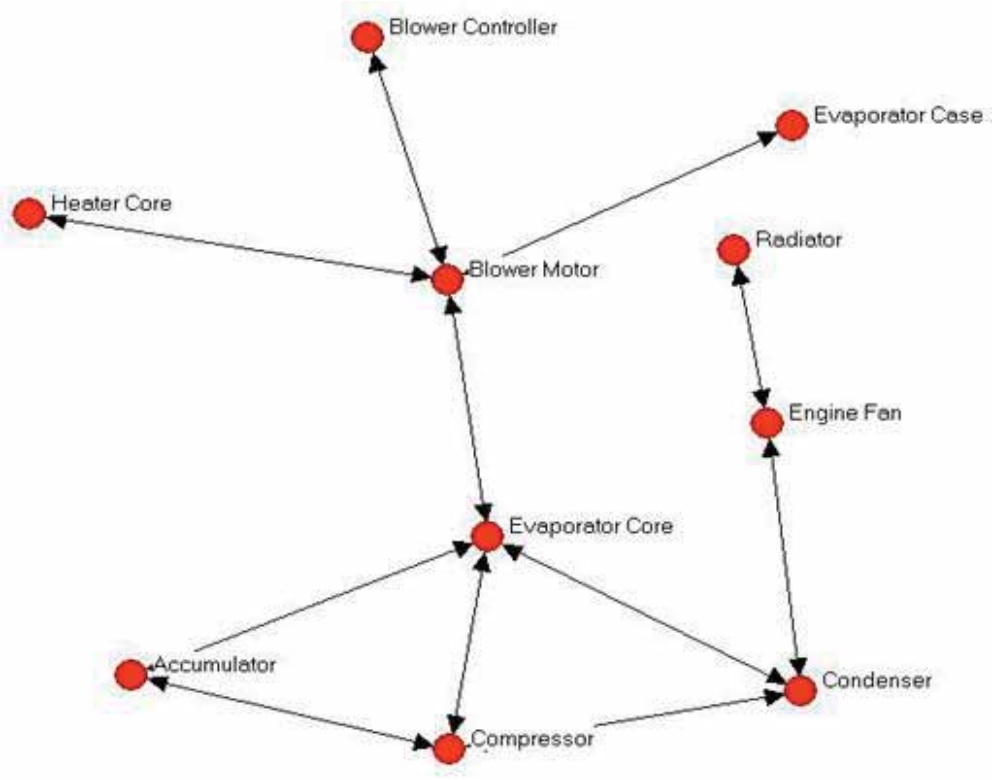

Figure 6 : Network Representation of Material design dependency.
Components not shown have no links with any other component.

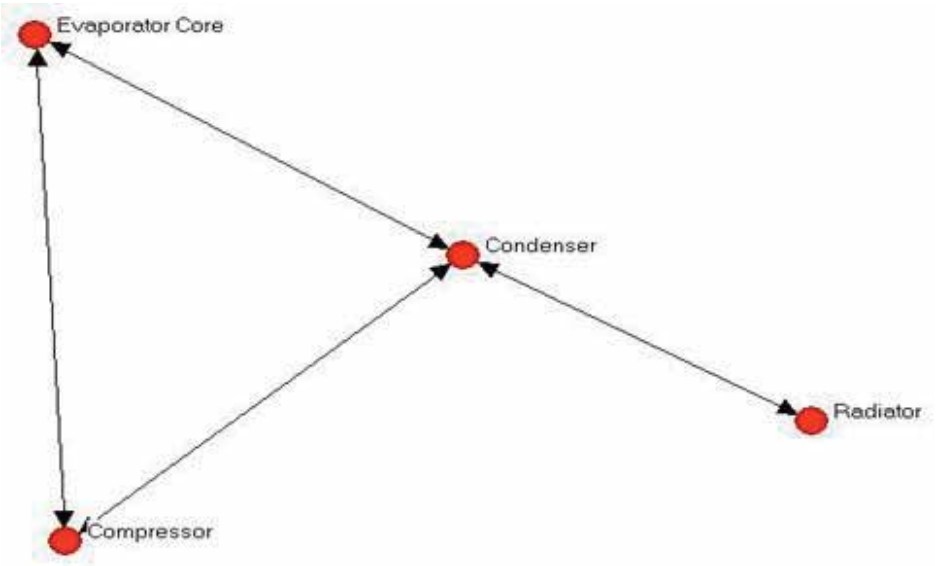

Figure 7 : Network Representation of Energy design dependency.
Components not shown have no links with any other component for this dependency. See Table 3.

#		Material Degree	Material Bridge	Energy Degree	Energy Bridge	Spatial Degree	Spatial Bridge	Info Degree	Info Bridge
1	EATC Controls	2	2	2	2	1.94	2	1.38	1.89
2	Refrigeration Controls	2	2	2	2	1.88	1.8	1.75	2
3	Heater Hoses	2	2	2	2	1.88	1.33	2	2
4	Command Distribution	2	2	2	2	1.5	1.98	2	2
5	Sensors	2	2	2	2	1.94	2	1.88	2
6	Radiator	1.88	2	1.88	2	1.75	2	2	2
7	Engine Fan	1.75	1.88	2	2	1.69	1.93	2	2
8	Condenser	1.63	1.93	1.63	1.5	1.63	1.88	2	2
9	Compressor	1.63	1.33	1.75	2	1.88	1.8	1.75	2
10	Accumulator	1.75	2	2	2	1.75	1.93	2	2
11	Evaporator Core	1.5	1.95	1.75	2	1.63	1.94	2	2
12	Heater Core	1.88	2	2	2	1.75	1.83	2	2
13	Blower Motor	1.5	1.95	2	2	1.69	1.69	2	2
14	Blower Controller	1.88	2	2	2	1.69	1.69	1.88	2
15	Evaporator Case	1.88	2	2	2	1.38	1.94	2	2
16	Actuators	2	2	2	2	1.81	1.69	1.88	2

Table 5 : The Modularity values for Climate Control System

Analysis of Climate Control System Values

Table 5 shows that the data values are aligned with the primary intuition developed via the network representations in Figure 6 and Figure 7. We see that partitioning the DSM into various design dependencies enriches the content and provides critical information regarding the overall product network. We also see that different components can be classified as more modular or less modular depending upon the design dependency being investigated. So, for information dependencies, the EATC controls are the most integral component, while for material dependencies, evaporator core and blower motor are the most integral (they have the lowest degree modularity values). Once again we see that the bridge modularity values provide added information, needed for day-to-day operations and policy level decisions. The bridge modularity values provide us with the 'sensitive' components – the compressor is a sensitive component from the material point of view, even though it is modular from a degree point of view. Hence, while the compressor material design is perhaps not critical, its operations affect the climate control system the most – therefore the performance parameters of the compressor are the most critical. The Compressor is our material 'hot-spot' - while condenser and heater hoses are similarly sensitive components with respect to energy and spatial dependencies.

Overall, we can interpret that components with low degree modularity values provide direct information about the modularity or integrality of that component while the low bridge modularity values provide information about the sensitive spots in the product. These results can then be applied on the Modularity Sourcing Matrix (Figure 5) to develop sourcing recommendations. For example, we can recommend that four components - evaporator core, blower motor, heater hoses and EATC controls are components which form the most integral components of the Climate Control System. The Climate Control unit firm may like to develop strong technical competencies on these four components, even if it does not wish to produce them in-house. Again, if the product life cycle of EATC control system is such that its obsolescence is high, then the focal firm may like to develop strong competencies around the EATC control system production. Additionally, we can recommend that enhanced quality control and design processes need to be deployed for the compressor, condenser, heater hoses and EATC controls, to ensure that their design and production (either in-house or outsourced) is robust. Inventory recommendations for production and service related issues can also be developed. For example, service centers may be advised to treat the compressor, the condenser, heater hoses and EATC controls as critical service items, items which must never be allowed to be out of stock. These four components, along with

evaporator core and blower motor are also the items which must be high on the training agenda of service personnel at the service centers of the automobile firm.

7. Conclusions and Future Work

This paper makes two important contributions. First, it enriches the product architecture literature by providing formal definitions and measures of modularity at the component level. *It shows that the Network Adjacency Matrix as understood in graph theory literature and the Design Structure Matrix as understood in the Product Architecture literature are similar in their treatment.* The paper takes a network approach based on graph theory to define two measures of component modularity. The two definitions of component modularity emphasize two different and vital aspects of modularity relevant at the component level. *Degree modularity* is related to the strength of design dependencies with adjacent components. *Bridge modularity* is the indicator of sensitivity of components. These measures are quantified and interpreted for two different products and the paper shows how design dependencies data can provide information about component modularity.

Second, the paper also illustrates how to use component modularity measures to develop day-to-day operational level as well as strategic level sourcing related recommendations by taking a moderating variable into consideration. The paper takes the variable of component obsolescence and develops the modularity sourcing matrix depending upon the level of obsolescence. Similar sourcing matrices can be developed for other procurement parameters. The paper also discusses how some of these parameters like inventory and after sales service can be related to modularity of components. The easy computation and use of modularity measures at the component level may make it easier for managers to develop sourcing recommendations.

Although the two proposed measures of component modularity enrich our understanding and are relatively simple to calculate (once the network of component design interfaces has been documented), future research may concentrate on the dynamic effects of these and alternative measures that capture architectural properties of components. How do these modularity measures change over time and as technologies change? An interesting question can also be related to fact of multiple use of same component. A component is modular or integral with respect to a product. But the same component may have a different modularity measure if it is also used in another product procured by the same firm. How can the sourcing decision matrix be developed for such a scenario? These and associated queries form future research questions for us.

8. Acknowledgments

I am grateful to Dr. Robert Stone for giving me permission to use the data available at the Design Engineering Lab. I am also grateful to Dr. Steven Eppinger for giving me permission to use the dataset described in Pimmler and Eppinger (1994). I appreciate the helpful comments provided by Dr. Manuel Sosa during the early parts of this research.

References

ALEXANDER, C. 1964, *Notes on the Synthesis of Form*. Harvard University Press, Cambridge, MA.

BALDWIN, C. Y., K. B. Clark, 2000. *Design rules: Volume 1: The power of modularity*, MIT Press, Cambridge, MA.

BARABÁSI, Albert-Laszlo, 2002. *Linked: The New Science of Networks,* NY: Perseus.

CLARKSON, P.J., Simons, C.S., and Eckert, C.M., Predicting change propagation in complex design. *ASME Journal of Mechanical Design,* 126(5), 765-797.

Design Repository, Design Engineering Lab, University of Missouri-Rolla, http://function.basiceng.umr.edu/

DIESTEL, Reinhard, Graph Theory, 2005. Springer Verlag, New York. Electronic Edition at http://www.math.uni-hamburg.de/home/diestel/books/graph.theory

DYER, Jeffrey H., 1996. Specialized Supplier Networks as a Source of Competitive Advantage: Evidence from the Auto Industry, *Strategic Management Journal*, 17(4).

ECKERT, C.M., Clarkson, P.J., and Zanker, W. 2004. Change and customization in complex engineering domains. *Research in Engineering Design*, 15(1), 1-21.

EPPINGER, S.D., D.E. Whitney, R.P. Smith, D.A. Gebala. 1994. A model-based method for organizing tasks in product development. *Research in Engineering Design*, 6(1), 1-13.

ERNST, R., Kamrad, B. ,2000. Theory and methodology: evaluation of supply chain structures through modularization and postponement, *European Journal of Operational Research,* 124, 495-510.

FINE, C.H., 1998. Clockspeed-Winning Industry Control in the Age of Temporary Advantage, Perseus Books, Reading, MA.

GARUD, R. and Kumaraswamy, A., 1993. Changing Competitive Dynamics in Network Industries: an Exploration of Sun Microsystem's Open Systems Strategy, *Strategic Management Journal,* 14, 351-369.

GERSHENSON, J.K., Prasad, G.J., and Allamneni, S., 1999. Modular product design: a life-cycle view. *Journal of Integrated Design and Process Science*, 3(4), 13–26.

GERSHENSON, J.K., G.J. Prasad, and Y. Zhang. 2004. Product Modularity: Measures and Methods. *Journal of Engineering Design,* 15(1), 33-51.

GERWIN, D. , Coordinating new product development in strategic alliances, 2004, *Academy of Management Review,* 29(2), 241-57.

HARARY, F. 1994. *Graph Theory*. Reading, MA: Adison-Wesley.

HENDERSON, R., K. Clark. 1990. Architectural innovation: the reconfiguration of existing product technologies and the failure of established firms. *Administrative Science Quarterly*, 35(1), 9-30.

JARRATT, T., Eckert, C. and Clarkson, P.J. ,2004. Development of a product model to support engineering change management, *TMCE 2004, Lausanne, Switzerland,* 1, 331-342.

KIRSCHMAN, C. F. and Fadel, G. M. 1998. Classifying Functions for Mechanical Design. *Journal of Mechanical Design*, 120, 475-482.

KRISHNAN, V., Ulrich, K.T., Product development decisions: a review of the literature, 2001, *Management Science*, 47(1), pp. 1-21.

LAU, A.K.W. and R.C.M. Yam, 2005. A case study of product modularization on supply chain design and coordination in Hong Kong and China, *Journal of Manufacturing Technology Management*, 16(4), 432-446

M. BENSAOU and Erin Anderson, 1999. Buyer-Supplier Relations in Industrial Markets: When Do Buyers Risk Making Idiosyncratic Investments?, *Organization Science*, 10(4).

MACCORMACK, A., Rusnak, J. and Baldwin, C. 2004. Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code. *HBS Working paper 05-016. Harvard Business School.*

MIKKOLA, J.H., Gassmann, O., 2003. Managing modularity of product architectures: toward an integrated theory, *IEEE transactions on Engineering Management*, 50(2), 204-218.

MIKKOLA, Juliana H., 2006. Capturing the Degree of Modularity Embedded in Product Architectures, *Journal of Product Innovation Management*, 23(2).

NEWCOMB, P.J., Bras, B., and Rosen, D.W., 1996. Implications of modularity on product design for the life cycle. *Proceedings of the 1996 ASME Design Engineering Technical Conferences*— 8th International Conference on Design Theory and Methodology (Irvine, CA).

NEWMAN, M. E. J., 2001. Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality, *Physical Review E*, 64, 016132.

NOVAK Sharon and Steven D. Eppinger, 2001. Sourcing By Design: Product Complexity and the Supply Chain, *Management Science,* 47(1).

ORTON, J.D. and Weick, K.E., 1990. Loosely coupled systems: A reconceptualization, *Academy of Management Review*, 15, 203-223.

PIMMLER Thomas U. and Steven D. Eppinger, 1994. Integration Analysis of Product Decompositions, *ASME Conference on Design Theory and Methodology*, Minneapolis, MN, pp. 343-351, September.

SALEH, J. H., 2005. Perspectives in Design: The Deacon's Masterpiece and the Hundred-Year Aircraft, Spacecraft, and Other Complex Engineering Systems, *Journal of Mechanical Design*, 127(3), 845-850.

SANCHEZ R. and Joseph T. Mahoney, 1996. Modularity, Flexibility, and Knowledge Management in Product and Organization Design, *Strategic Management Journal*, 17.

SCHILLING, M.A., 2000. Toward a general modular systems theory and its application to interfirm product modularity, *Academy of Management Review*, 25 (2), 312- 334.

SHARMAN and Yassine, 2004. Characterizing Complex Products Architectures. *Systems Engineering*, 7(1), 35-60.

SIMON, H.A., 1981. *The Sciences of the Artificial.* Second edition, MIT Press, Cambridge, MA.

SOSA, M.E., S.D. Eppinger, C.M. Rowles, 2003. Identifying modular and integrative systems and their impact on design team interactions. *Journal of Mechanical Design*, 125(2), 240-252.

STEWARD, D., 1981. The design structure matrix: a method for managing the design of complex systems, IEEE Transactions on *Engineering Management*, EM-28(3), 71-74.

SUH, N. P., 2001. *Axiomatic Design: Advances and Application*, Oxford University Press, New York.

SUH, Nam P., 1990. *The Principles of Design*, Oxford University Press, New York.

ULRICH, K. T. and S.D. Eppinger, 2004. *Product Design and Development.* 3rd. Edition. McGraw Hill, New York.

ULRICH, K.T., 1995. The role of product architecture in the manufacturing firm, *Research Policy,* 24, 419-440.

ULRICH, K. T. and David Ellison, 2005. Beyond make-buy: Internalization and integration of design and production. *Working paper, Wharton School, University of Pennsylvania, Philadelphia.*