



A method to evaluate direct and indirect design dependencies between components in a product architecture

Sangjin Jung¹ · Oyku Asikoglu² · Timothy W. Simpson¹

Received: 16 July 2016 / Revised: 3 March 2018 / Accepted: 4 June 2018 / Published online: 19 June 2018
© Springer-Verlag London Ltd., part of Springer Nature 2018

Abstract

Methods for evaluating the strength of design dependencies in a product architecture have been widely studied in the literature; however, evaluating the effects of direct and indirect interactions between components/modules remains a challenge. In fact, indirect connections between components/modules are often overlooked in many cases when evaluating design dependencies. Having a more consistent way of defining a product architecture that considers both its direct and indirect connections is important, especially when analyzing redesign complexity and change propagation. In this study, we propose a systematic method to evaluate direct and indirect design dependencies between components in product architectures. Interfaces are classified into six different types based on a thorough review of the literature, and a method for evaluating design dependencies is introduced to estimate the relative importance of interfaces directly from a set of comparable products. Using an electrical circuit analogy, the proposed method can quantify both direct and indirect design dependencies between components within a product architecture. We compare design dependency results for different wireless computer mice to validate the effectiveness of the proposed method. The results show that using the proposed design dependency measure including direct and indirect effects provides more reliable design dependency results.

Keywords Design dependency · Interface · Product architecture · Design structure matrix · Electrical circuit analogy · Change propagation

1 Introduction

Design dependencies affect many of the important architectural decisions of a product. Even in modular structures, managing interfaces becomes much more difficult as the product becomes more complex when new features are added. Due to the amount and different types of connections that they may contain, some architectures might be more susceptible to amplification of design changes within the system than others (Eckert et al. 2004; Jarratt et al. 2011). Such architectures are most likely to increase the cost and duration of the redesign process (Jarratt et al. 2011); hence, it is important to be able to determine the impact of design dependencies in advance.

Methods for quantifying design dependencies between components/modules in a product architecture have been widely studied in the literature, and many evaluate the strength of design dependencies for direct and/or indirect connections based on engineers' experience and knowledge (Hamraz et al. 2015; Koh et al. 2012; Sosa et al. 2007). Engineering intuition and understanding of product architectures are essential to define, analyze, and improve a product's architecture, but using too much subjective information may result in biases when assessing design dependencies (Eckert et al. 2004; Hölttä and Otto 2005). To reduce bias when evaluating product architectures and design dependencies, recent studies by Moullec et al. (2013), Hamraz et al. (2013), and Ye et al. (2015) have proposed the integration of information on uncertainties related to interfaces and components, but the subjective input data should still be carefully considered. More importantly, many methods only consider direct connections between components/modules when evaluating design dependencies (Dobberfuhr and Lange 2009; Martin and Ishii 2002; Pimmler and Eppinger 1994; Tilstra et al. 2012), and we may overlook important indirect connections

✉ Timothy W. Simpson
tws8@psu.edu

¹ The Harold and Inge Marcus Department of Industrial and Manufacturing Engineering, The Pennsylvania State University, University Park, PA 16802, USA

² QAD, Inc., Mt. Laurel, NJ 08054, USA

in product architectures. This underestimates the effects of a design change that propagates through both direct and indirect connections within an architecture (Eckert et al. 2004). As the complexity of the overall system increases, the difficulty of accounting for indirect interactions also increases.

This study proposes a systematic procedure to define design interfaces and evaluate corresponding design dependencies based on data extracted directly from a set of comparable products. In this work, a weighted design dependency measure using an electrical circuit analogy is introduced to consider direct and indirect connections between components. Related research on interface representations and design dependency measure is reviewed in the next section. Section 3 describes the new design dependency measure in detail. Section 4 presents a case study for the weight calculation based on data collected from 21 electro-mechanical household products. The procedure to evaluate change resistance is also introduced in this section. In Sect. 5, the architectures for wireless computer mice belonging to different generations are compared, and the effectiveness of the design dependency measure including indirect interaction effects is demonstrated. Section 6 presents closing remarks and future work.

2 Review of related research

2.1 Structural representation approach

Methods for product architecture and interface representation have been widely studied (Browning 2001). Many representation methods have focused on (1) functional approaches to describing product architectures (Hirtz et al. 2002; Hundal 1990; Pahl et al. 2007) and (2) structural approaches (Dobberfuhr and Lange 2009; Sanchez 1994) using networks, matrices, etc. such as Design Structure Matrices (DSMs) (Browning 2001; Eppinger and Browning 2012; Steward 1981). It is well known that structural approaches effectively describe and can be used to analyze structural interfaces and connection strengths between components. A typical structural representation using a DSM maps the relationships between components with a binary representation (Browning 2001). The existence of a relationship is typically represented with a dot or a number “1”, whereas cells for non-existing relationships are left blank or set to “0”. This approach minimizes bias on deciding whether an interface exists between components or modules, because it is easy to determine when components are, or are not, connected to one another; however, it does not discriminate different dependency strengths or different types of interfaces.

The first explicit structural analysis of interface classification was performed by Sanchez (1994). He identifies five different types of interfaces: (1) attachment, (2) transfer,

(3) control and communication, (4) spatial, and (5) environmental interfaces. Around the same time, Pimmler and Eppinger (1994) classified interactions between components into four categories: (1) spatial, (2) energy, (3) information, and (4) material. Even though some interaction definitions such as spatial and information interfaces correspond one-to-one to the definitions of *spatial* and *control and communications* interfaces in Sanchez’s study, other descriptions show slight differences from his work. Unlike Sanchez, Pimmler and Eppinger’s method does not include structural connections or environmental effects between connections explicitly in their work. In addition, Sanchez (1994) describes energy or material exchange between components under the same category (transfer interfaces), while Pimmler and Eppinger (1994) distinguish the two by separating them into different interface categories called energy and material interactions.

Taking a step further from Sanchez’s work, Pimmler and Eppinger (1994) also propose a method to quantify the importance of interactions in an architecture. They acknowledge that some connections can be more important compared to others and that some connections are more desirable, while some others might be detrimental. They employ a rating method to assign weights to connections on a five-point scale ranging from +2 to –2. Sosa et al. (2003) extend their work by adding a structural interface type, which defines the requirement of transferring loads or containment between components. They categorize the interface types into two major groups: (1) spatial-type and (2) transfer-type interfaces. Spatial-type interfaces only include spatial relations between components, while transfer-type interfaces include all the rest of the interface types except information interfaces. Similar to Pimmler and Eppinger (1994), Sosa et al. (2007) also utilize the rating method to assign weights from +2 to –2. Ko (2013) suggests a similar taxonomy of interface types (i.e., geometry, energy, signal, and material) and utilizes a fuzzy set theory to measure dependency strengths between components, but detailed definitions of each interface type are not provided.

The next study that considers defining structural connections within the product architecture is Ericsson and Erixon (1999)’s work on Modular Function Deployment. As part of their approach, they define different types of interfaces such as fixed, moving, and media-transmitting interfaces. Fixed interfaces are used to physically connect modules and transfer forces, whereas moving interfaces are used to transfer rotating or alternative energy. Media interfaces are used to define the transmission of fluids or electricity between modules. Even though there are several different types of connections within these categories, only two of the defined interface types are used throughout their study, namely, geometry and energy.

Bettig and Gershenson (2006) introduced a more formal module interface representation and defined several different interface types: attachment, transfer, control and communications, power (i.e., electrical), spatial, field, and environmental interfaces. They ultimately narrow down their list to four types: (1) attachment, (2) control and communications, (3) transfer, and (4) field interfaces, but they are not explicit about how they arrived at this reduced set of types.

Jarratt et al. (2004) introduce a list of “linkage” definitions that represent structural connections in the system. A team of design engineers are chosen to determine the linkages in the system. After initial suggestions of mechanical, spatial, thermal and electrical linkages, the design team creates their own comprehensive list of linkages. Their list includes eight different connections: (1) mechanical steady state, (2) mechanical dynamic, (3) spatial, (4) thermal steady state, (5) thermal dynamic, (6) electrical signal, (7) electrical earth, and (8) electrical dynamic.

Tilstra et al. (2009) propose interface DSMs to represent product architectures. They use their High Definition-DSM model to compare the flexibility between different products, where they use different layers for each of their interaction types to represent the product architecture. Even though the method provides detailed information about product interactions, one needs to apply many steps to construct the HD-DSM of the analyzed product. Considering the necessity of creating the “black box” models for each element and HD-DSMs for each different interaction in each subsystem, the method becomes a daunting task even for a small electro-mechanical product such as the electrical screwdriver provided as the case study (Tilstra et al. 2009).

Dobberfuhr and Lange (2009) assert that the complexity of connections increases as the variety of connections between two components increase, and they define seven different types of interfaces: (1) attachment, (2) transfer, (3) spatial, (4) command and control, (5) field, (6) environmental, and (7) user interfaces. An interface has a different level of complexity, because of single, double, triple, etc. interface combinations that exist within the product architecture. The complexity level is referred to as the Module Complexity Score (MCS), which is computed as follows:

$$MCS = n_1 + 2n_2 + 3n_3 + 4n_4 + 5n_5 + 6n_6 + 7n_7, \quad (1)$$

where n_1, n_2, n_3 , etc. represent the number of interface types at complexity level one, two, three, etc. respectively. For example, if an interface between components consists of a combination of attachment and transfer interfaces, then the value of MCS is 4, because the number of interface types at complexity level two, n_2 , is 2, which is then multiplied by 2. However, this method assumes that all different kinds of interface types have the same importance (e.g., an attachment interface is as complex as a command and control

interface). In addition, indirect connections within the system are mentioned in their study but are not evaluated.

Ariyo et al. (2010) define their interface classification based on the previous studies of Jarratt et al. (2004) and Lockledge and Salustri (1999). The interface information provided in their study lists four types of interfaces: (1) mechanical steady state, (2) spatial, (3) electrical signal, and (4) air flow; however, the limited description of their interface types points to a case-specific definition. Consequently, their study fails to provide a standardized interface classification method for use in other products.

Figure 1 summarizes structural representation approaches for interfaces and shows the characteristics of defined interface types. Most of the approaches have partially similar types of interfaces; for example, spatial- and transfer-related interfaces can be considered in common. On the other hand, most of them also have case-specific types of interfaces (e.g., airflow interface, user interface, etc.); so they might be impractical for frequent use in other types of products. In addition, most of the methods can take into account only direct connections between components without considering indirect connections in product architectures. There are several studies focused on evaluating the effects of indirect connections in product architectures. The research related to change propagation and design dependencies considering indirect connections is reviewed in the next section.

2.2 Change propagation in product architecture

Interest in the effects of design changes in product architectures started increasing as an extension to the research on product platform architectures and design for variety (Martin and Ishii 2002). One of the most prominent works on the characterization of the engineering design propagation is done by Eckert et al. (2004), in which they identify the problems of grasping the relationships of a complete system as “a major source of emergent problems”. Their study defines a complex product as a product with closely linked parts and systems, where a system change most likely transfers to other systems. They examine a complex system of a helicopter and their interviews with the engineering team determines that only 70% of the whole system was completely understood (Eckert et al. 2004). The lack of understanding the complete system comprehensively also brings major downfalls in determining how design changes propagate through a system. They elaborate on the sources of change, system problems, change propagation and types of components behavior under change propagation, change prediction and consequences of change has been described under the example of helicopter design and some shortcomings of relying on engineering intuition for design change are also pointed out (Eckert et al. 2004).

Sanchez, 1994	Pimmler and Eppinger, 1994	Ericsson and Erixon, 1999	Sosa et al., 2003	Jarrat et al., 2004	Bettig and Gershenson, 2006	Tijkstra et al., 2009	Dobberfuhl and Lange, 2009	Ariyo et al., 2010
Attachment Interfaces: structural connections	Spatial Interactions: adjacency or orientation between two elements	Fixed Interfaces: only connect modules and transmit forces	Structural: indicates a requirement related to transferring loads, or containment	Mechanical Steady State (Ms): relations of physical contact	Attachment Interfaces: mechanical connection to restrict relative motion and transmit force	Movement: Translational (LRM) and Rotational (RRM)	Attachment Interfaces (A): physical connections	Mechanical Steady State (M): relations of physical contact
Spatial Interfaces: geometrical and locational constraints of a component in the overall arrangement of other components	Material Interactions: material exchange	Moving Interfaces: transmits energy in forms of rotating or alternating forces	Spatial: indicates a requirement related to physical adjacency for alignment, orientation, serviceability, assembly or weight	Mechanical Dynamic (Md): interactions through a fluctuating force or displacement	Transfer Interfaces: intends to convey energy, material, or signal through mechanical means	Spatial: Proximity (P) and Alignment (A)	Spatial Interfaces (S): module boundaries	Spatial (S): important interactions of adjacency and orientation
Transfer Interfaces: the flow of materials or power between components	Energy Interactions: energy transfer between two elements	Media Interfaces: fluids or electricity	Material: indicates a requirement related to transferring airflow, oil, fuel or water	Spartial (S): important interactions of adjacency and orientation	Control and Power Interfaces: transmits electrical power or signal	Material: Human (HM), Gas (GM), Liquid (LM), Solid (SM), Plasma (PM), Mixture (MM)	Transfer Interfaces (T): power or media transfer	Airflow (A): material (air) exchange between components
Control and Communications Interfaces: the relation of signal or information flow between two components	Information Interactions: information or signal exchange		Energy: indicates a requirement related to transferring heat energy, vibration energy, electrical energy or noise	Thermal Steady State (Ts): steady state temperature difference	Field Interfaces: transmits energy, material, or signal as an unintended side effect of a module	Energy: Human (HE), Acoustic (AE), Biological (BE), Chemical (CE), Electrical (EE), Electromagnetic (EME), Hydraulic (HYE), Mechanical (ME), Magnetic (MAG), Pneumatic (PE), Radioactive (NE), Thermal (TE), Strain Energy (SE)	Command and Control Interfaces (C): the state of communication and/or controls between modules	Electrical Signal (E): relations of signal transfer
Environmental Interfaces: the effect of weight, sound, vapor, corrasions, heat, radiation, electromagnetic, etc. between two components				Thermal Dynamic (Td): fluctuating temperature difference		Information: Status (SI) and Control (CI)	Field Interfaces (F): the functioning of one component can generate heat, magnetic fields, vibrations or other	
				Electrical Signal (Es): relations of signal transfer			User Interfaces (U): human interaction with features within a module system	
				Electrical Earth (Ee): electrical earth connection			Environmental Interface (E): effect of ambient conditions	
				Electrical Dynamic (Ed): logic driven behavior				

Fig. 1 Summary of structural representation approaches for interfaces

In a follow-up study, Clarkson et al. (2004) introduce a method to predict change propagation using DSMs. They describe change relationships as a combination of likelihood and impact. Likelihood is defined as the probability of a design change requiring a change in the product architecture and impact is defined as the average proportion of the redesign if the change propagates in the system. The data needed for the construction of the likelihood and impact matrices are collected from the historic information on previous design changes and engineering expertise (Clarkson et al. 2004), which incorporates personal bias into the analysis. Even though they acknowledge the importance and try to capture the effects of indirect connections in the system using “propagation trees”, their method fails to provide a practical solution to be used in products with lower level system granularity in which there are increased numbers of components. Later on, the method of using “propagation trees” is formalized into the Change Prediction Method (CPM) to visualize direct and indirect change propagation in computer software (Keller et al. 2005).

Suh et al. (2007) also focus on managing change propagation in systems providing a change propagation index that enables classifying components based on their change transmittance characteristics such as constants, multipliers, carriers and absorbers defined by Eckert et al. (2004). The work provides a detailed look into the mechanism of change

propagation and extends the work to address of cost determination and uncertainty analysis. However, the construction of the change propagation matrix heavily depends on observations that increase the risk of incomplete or incorrect data.

Building on the studies of Suh et al. (2007) and Eckert et al. (2004), Giffin et al. (2009) introduce a network-based analysis technique that can be applicable to large data sets. The study defines change propagation characteristics of the system and uses a DSM to visualize the connections in terms of physical connections with information and energy flows. Even though the study is comprehensive, it focuses on the complex network system composed of software, hardware and documentation areas.

In recent years, change propagation approaches have focused primarily on assessing change propagation in multi-domains. Ahmad et al. (2013) summarize existing approaches and classify domains of change propagation into requirements, functions, function form, components, parameters, documents, tasks, people/agents, and events. They also introduce a cross-domain approach for assessing change impacts considering the information domains of requirements, functions, components, and design process. Pasqual and de Weck (2012) also developed a multi-layer network model for analyzing change propagation by composing three layers: (1) product layer, (2) change layer, and (3) social layer. Based on CPM (Clarkson et al. 2004), Koh

et al. (2012) suggest a multiple-domain matrix to consider the dependencies within/between different domains (i.e., components, change options, and product requirements). Similar to Koh et al.'s (2012) work, CPM has also been applied to function–behavior–structure linkage models (Hamraz et al. 2012, 2015; Hamraz and Clarkson 2015). The approaches using CPM (Ahmad et al. 2013; Hamraz et al. 2012, 2015; Hamraz and Clarkson 2015; Koh et al. 2012) require input data by users to define change likelihood values in their multi-domain matrices, which may be subjective in nature. The multi-domain matrix suggested by Koh et al. (2012) also uses qualitative information to create the House of Quality (HoQ). Thus, even though change propagation approaches have been extended to assess change propagation in multi-domains, most of the approaches utilize qualitative or subjective information to capture design dependencies and establish change propagation models.

This section reviewed key studies related to the definition and quantification of flexible product designs and change propagation. While methods have been developed to assess change propagation and complexity in product architectures, a method that provides a more quantitative approach to evaluate the design dependencies within product architectures is absent. In the next section, we introduce a modified classification of interface types and a systematic method to evaluate direct and indirect design dependencies between components in a product architecture.

3 Design dependency measure

3.1 Definition of interface types

Accurate definitions of interface types and importance are fundamental for a correct understanding of the product's architecture. The interface data also affects quantifying the impact of design dependencies (Hamraz et al. 2013; Sosa et al. 2007). As discussed in Sect. 2.1, structural representations for interfaces vary in the literature, but the interface categorizations in existing approaches are product-specific or impractical for frequent use (e.g., the spatial interface (Pimmler and Eppinger 1994) and transfer interface (Dobberfuhr and Lange 2009) by themselves may be too general, while the airflow interface (Ariyo et al. 2010) may be too product-specific). Building on the studies described in the previous section, a new classification of interfaces is introduced in this study. Our approach classifies interfaces into six different types of interfaces: (1) attachment, (2) spatial, (3) power, (4) control and communication, (5) transfer, and (6) field. The definitions of the different types of interfaces are described in Table 1.

The first main difference of this classification from previous work is in the physical connections. Pimmler and

Eppinger (1994) identify physical and geometrical relations with a single interface type, whereas Sanchez (1994) and Dobberfuhr and Lange (2009) identify spatial interfaces as geometric relations and attachment interfaces as the physical connections between components. The differences between two different types of connections are not clear in either of their studies. Our proposed interface classification clearly differentiates between the two based on the need for an additional connector (e.g., bolt, screw, or rivet). Spatial interfaces represent two different relations between components. The first one is dimension relation (i.e., the size and fit of components within the architecture), and the second facilitates a physical connection without the need of an additional connector.

Another main difference in the proposed classification is electrical connections. While previous studies (Dobberfuhr and Lange 2009; Sanchez 1994) include electrical connections in transfer interfaces, this work assigns electrical connections to a separate grouping. The use of the electrical connections is more frequent compared to other transfer interfaces, and they also differ in terms of required assembly procedures. Lastly, we omit the use of the user interface type, which is defined by Dobberfuhr and Lange (2009), since this study only focuses on the design dependencies between components in a product architecture.

3.2 Quantifying design dependency

Considering all the different possibilities of interactions between components, one can quickly realize that not all connections have the same intensity; therefore, not all the connections have the same design dependency. This work introduces a novel approach to calculate the strength of design dependencies using connection information contained in a DSM. The relative strength of connections between components is first calculated using the weighted Module Complexity Score (wMCS). Unlike MCS in Eq. (1), the proposed measure allows each interface type to have different weights. The resulting wMCS formulation is given in Eq. (2):

$$\text{wMCS} = \sum w_i (n_{1i} + 2n_{2i} + 3n_{3i} + 4n_{4i} + 5n_{5i} + 6n_{6i}), \quad (2)$$

where w_i = weight of interface type and $i = A, S, P, C, T$ or F .

The relative weights of different types of interactions are determined based on the frequency of occurrence of the different types of interfaces using a weighting function. Using the non-linear weighting function given in Eq. (3), weights that range between 1 and 10 are assigned to different interface types based on their frequency of occurrence. A non-linear weight function is preferred for calculations, since it captures the nature of the input data (i.e., frequency of occurrences of interface types). Here, $w(\alpha)$ represents a

Table 1 Definition of interface types

Interface type	Notation	Definition
Attachment	A	the structural connections between two components that require a type of connector (e.g., bolts, screws, rivets)
Spatial	S	geometrical and locational constraints of a component with respect to other components
Power	P	the electrical connection between two components unlike communications and controls interfaces
Control and communications	C	signal or information flow between two components in which the state of one component is communicated or controlled by another component
Transfer	T	the flow of materials or power between components, (e.g., water flow in a coffee maker, transfers of motion such as torque)
Field	F	the interaction between two components in which one component can generate heat, vibration, or magnetic field

weighting function normalizing the statistical information by the frequency of occurrence α :

$$w(\alpha) = C_1 e^{-C_2 \alpha} + C_1 C_2 e^{-C_2 \alpha}, \quad (3)$$

where constants C_1 and C_2 are estimated from the desired sensitivity that characterizes the interface type–occurrence relationship. The interface type–occurrence relationship suggests that more commonly employed interfaces are preferable over the less frequently used ones. This is because more commonly used interfaces are far more likely to have compatibility benefits in component’s development, manufacturing, assembly, etc.

Meanwhile, calculating the cumulative effect of direct and indirect connections in the product architectures brings computational difficulties. To overcome these difficulties, we model the product architecture as an electrical circuit, where the connections between components are represented by resistors. Electrical components work in accordance with basic circuit theory. These elements define the relationships between fundamental electrical parameters such as current and voltage. This study aims to extend the electrical circuit analogy to product design and show that it is not only applicable but also beneficial in terms of calculating design dependencies in product architectures.

Other disciplines such as computer science and social networks (Bozzo and Franceschet 2013; Klein and Randić 1993) also use analogies of circuit theory (DeCarlo and Lin 1995), in which “circuit elements” define relationships between other fundamental parameters. For example, Stephenson and Zelen (1989) introduced a model of centrality based on information distances among nodes in a network, and Parraguez (2015) extended their work to compute the information centrality nearness matrix of actual process architectures. Klein and Randić (1993) and Bozzo and Franceschet (2013) define *resistance distance* using an electrical network theory to compute the information distance directly and/or indirectly connected between nodes. Klein and Randić (1993) mathematically prove that the resistance function on a graph can be a distance function. According to their definition, the lower resistance distance means that the

information contained in direct and indirect paths between two nodes is higher. In our approach, however, the low equivalent resistance between two components means high design dependency between them. The detailed procedure to quantify design dependencies using the electrical circuit theory follows.

Similar to an electrical circuit, the components within a product also form a network, wherein all components are connected to each other directly or indirectly. We consider components having relationships with other components of different strengths. Unlike most DSMs, where interactions are represented in a binary nature, this study uses an interface-based DSM, where connections have calculable values. This shift in perspective is reflected by the analogy of an electrical circuit, where wMCS values represent the concept of electrical conductance. This approach proposes that, as an interface between two components becomes more complex (i.e., as the wMCS value increases), its ability to “conduct” or “transmit a design change” also increases much like conductance in an electrical circuit.

The electrical circuit analogy can thus capture the amount of “change propagation” throughout a product architecture. In an electrical circuit, the flow of current is dependent on the resistance within the circuit. Similarly, change propagation in a product architecture is related to the design dependencies between components. This work defines design dependencies as characteristics of the interfaces between components. Therefore, a design change in a product architecture is considered as “current” flowing through the electrical circuit. Similar to an electrical circuit in which the resistance of a resistor defines the amount of current passing through it, the characteristics of the connections between components (i.e., interfaces) determine the amount of change transmitting throughout the product architecture.

If we refer to Ohm’s Law, $I = V/R$, our analogy casts I as a design change, V as the redesign effort, and R as the design dependency between components. In this analogy, the reciprocal of the wMCS values (referred to as *change resistance* in our analogy) are used as resistance values. If there exists a high design dependency (i.e., small

resistance value) between Components A and B, when the design of the Component A is changed, then Component B may be affected a lot. By simulating the product architecture as electrical circuits and running the models with given change-resistance scores, the impedance results for any chosen connection are calculated. These impedance calculations represent the overall design dependency for a given interface in the system by incorporating all of its direct and indirect connections. The electrical circuit model for impedance calculation can be easily created using various software such as the MATLAB/Simulink software package (MathWorks 2015).

For example, as shown in Fig. 2 when Components A, B, and C are linked each other within a product architecture, the resultant change-resistance between Components A and B, CR_{AB} is computed as

$$CR_{AB} = \frac{1}{\frac{1}{CR_{p,AB}} + \frac{1}{CR_{p,AC} + CR_{p,BC}}} \quad \text{where } CR_{p,ij} = \frac{1}{wMCS_{ij}}, \quad (4)$$

where $CR_{p,ij}$ is the preliminary change-resistance between components i and j (i.e., the reciprocal value of $wMCS_{ij}$). CR_p is used to construct an electrical circuit model, and CR_p represents only the direct design dependency between components without considering the effects of indirect connections in the DSM. In Eq. (4), CR_{ij} is the final change-resistance (i.e., design dependency) including the effects of direct and indirect connections.

Figure 3 shows an electrical circuit model created using the MATLAB/Simulink software to compute design dependencies for this example. As seen in the figure, the connections between components are represented as resistors, while the actual components act like junction points for connections. To be able to calculate the resultant resistance between two components, the nodes in the model need to be connected to points of interest, and then the model can automatically display the resultant resistance between those two points in the display box, as shown in Fig. 3.

The proposed method evaluates design dependencies based on data extracted directly from a set of existing products (e.g., similar types of products in the market). Therefore, designers can reduce their subjective judgements needed to assess design dependencies; so, this method provides a less biased measure that can be used to compare different products designed by different engineers. In addition, electrical circuit models created in existing software such as MATLAB/Simulink can be utilized to compute direct and indirect design dependencies without implementing any additional algorithms. In the next section, a case study for small/mid-size electro-mechanical household products is presented to illustrate how to calculate interface weights

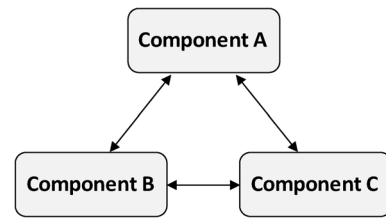


Fig. 2 Components and their connections

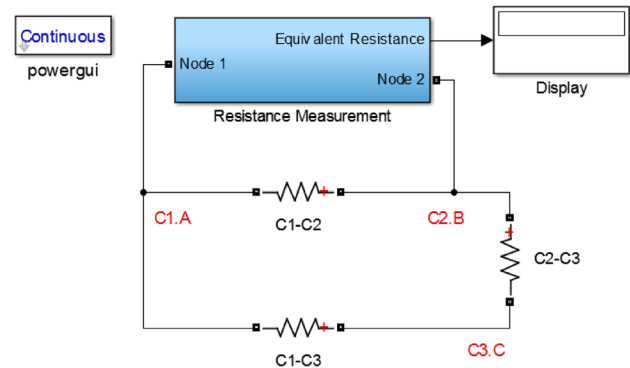


Fig. 3 Electrical circuit model

and change resistances in detail and to demonstrate the effectiveness of the proposed design dependency measure.

4 Case study: small/mid-size electro-mechanical household products

In this section, the calculation of design dependencies between components is described using a set of small to mid-size electro-mechanical devices and products. The procedure for evaluating design dependencies is summarized as follows:

- Step 1 Dissect a set of products and collect component and interface data.
- Step 2 Assign weights for each interface type.
- Step 3 Define interface DSMs and compute $wMCS$ values for each connection.
- Step 4 Build electrical circuit model and calculate change resistances.

The method begins by data collection from a representative set of comparable products. The data on components and interfaces are collected from actual products by disassembly in this example; a company, on the other hand, could easily maintain a database of this type of information for the products they develop. The level of dissection should be determined first and held constant throughout the process

to ensure consistency of the analysis. In some cases, the disassembly of a part might become extremely difficult. Electronic parts such as printed circuit boards are such an example and can be considered as a single component to simplify the process. Not listing connecting elements (e.g., nuts, bolts, screws, and cables) as components is also suggested to keep the component matrix at a manageable size and to avoid duplications in the data. Designers' judgement will influence Step 2 (e.g., when determining the weights C_1 and C_2); however, no additional subjective information is needed to define each interface type or strength and evaluate the impact of design dependencies in product architectures beyond that step.

In this work, we selected, dissected, and analyzed a set of 21 electro-mechanical household products that span the set of interfaces that we have defined (see Table 2). These 21 products include: Durabuilt® flashlight, sander, circular saw, and jigsaw, Black & Decker® DR260B drill, Aroma® ARC-150SB rice cooker, Rival® CKRVRCMO63 rice cooker, GE Rotisserie 168947 oven, GPX® C208B radio clock, Revlon® RVSP3505B1 electric brush, See 'n Say® The Farmer Says, Kodak® Fun Saver single-use camera, Logitech® M310 computer mouse, PowerShot® 5700M stapler, 5-Speed 100-W Toast Master® Hand mixer, Mr. Coffee® TF13 coffee maker, Rival® lightweight iron, Conair® 209GWP hairdryer, Honeywell® HZ-220 room heater, Conair® PR5007w phone, and Black & Decker® CHV1408 Dustbuster. Relatively few components and ease of dissection were important factors when choosing this set of products. These characteristics provided consistency during the data collection phase. All of the selected products have integral architectures with an average of seventeen components and can be represented using a DSM. As shown in Table 2, the selected 21 products comprise all of the different types of interfaces defined in the study. Based on the frequency of occurrence of these interface types, their relative weights are calculated.

Frequency of occurrence statistics are collected during the product dissection phase, where all the products are disassembled to the individual component level, and the connections between components are recorded in a component-based DSM. Based on the collected data, spatial interfaces are the most prominent type of interface in product architectures with an occurrence rate of 51%. The second most prominent type of interface is attachment interface: 34% of all interfaces in the analysis set are comprised of attachment interfaces. Electrical interfaces make up 10% of all interfaces, communications, and control interfaces represent 3%, and transfer and field interfaces comprise 1%. The respective occurrence ratios are plotted in the pie chart in Fig. 4.

In Step 2, the relative weights of different types of interfaces can now be calculated based on the data in Fig. 4

using the weighting function in Eq. (3). Figure 5 demonstrates the proposed weighting function and its corresponding sensitivity. The constants C_1 and C_2 were chosen such that the sensitivity approaches zero as the likelihood of occurrence increases. For instance, interfaces that occur frequently such as attachment and spatial interfaces will have similar and relatively small weights compared to interfaces that occur less frequently (e.g., transfer or field). Given the frequency of occurrence and increased sensitivity towards the less frequent interface types, individual weights for interface types have been determined. The final weights are shown in Table 3.

As explained in Table 1, there are six types of interfaces, and each connection between components might have:

- a single type of interface that only occurs once;
- more than one type of interface such as A(3) (i.e., three attachments) or PC; or
- a combination of interfaces such as ST or SCP.

In Step 3, the connection values for different types of interfaces are determined using the wMCS formulation in Eq. (2). Table 4 includes the weights for all kinds of combinations of interfaces. These weights can be utilized to compute design dependencies for the products that are comparable to the 21 products selected in this study.

Next, we can create electrical circuit models and then compute design dependencies in the product architectures. To describe the detailed procedure to evaluate design dependency, the DSMs and electrical circuit model for Microsoft Wireless Computer Mouse 1000¹ released in 2010 are created, as shown in Figs. 6, 7, 8, and 9. Figure 6 shows the interfaces between components in the computer mouse architecture. The interface representation in this figure is similar to a product component-design structure matrix (PC-DSM) suggested by Jankovic et al. (2012). Jankovic et al. (2012) and Holley et al. (2014) integrated the ontologies of interface types into PC-DSMs, but the PC-DSMs are not employed to quantify direct and indirect design dependencies. In this work, after representing the types of interfaces in Fig. 6, wMCS values of each interface are computed using the weights in Table 4 and Eq. (2) (see Fig. 7).

In Fig. 8, we create an electrical circuit model of the computer mouse DSM, and then the change resistances are finally computed, as shown in Fig. 9. For example, Fig. 8 shows an example of measuring an equivalent resistance value between the components C2.UH and C3.LH [i.e.,

¹ For more information, visit: <http://www.microsoft.com/accessories/en-us/mice>.

Table 2 Types of interfaces in analyzed products

Product	# of components	# of connections	Average connection per component	A	S	P	C	T	F
Aroma rice cooker	15	44	2.933	●	●	●	●	–	●
Circular saw	19	66	3.474	●	●	●	●	●	–
Coffee maker	16	54	3.375	●	●	●	●	●	●
Drill	16	86	5.375	●	●	●	–	●	●
Dustbuster	19	80	4.211	●	●	●	●	●	–
Electric brush	13	44	3.385	●	●	●	●	●	–
Flashlight	15	58	3.867	●	●	●	●	–	–
Hairdryer	28	96	3.429	●	●	●	●	–	●
Handmixer	10	38	3.800	●	●	●	–	●	–
Heater	25	106	4.240	●	●	●	●	–	●
Iron	13	62	4.769	●	●	●	–	●	●
Jigsaw	22	88	4.000	●	●	●	●	●	–
Mouse	11	40	3.636	●	●	●	●	–	●
Oven	28	114	4.071	●	●	●	●	●	–
Phone	16	54	3.375	●	●	●	●	–	–
Radio clock	15	64	4.267	●	●	●	●	–	●
Rival rick cooker	16	54	3.375	●	●	●	●	–	●
Sander	15	46	3.067	●	●	●	●	●	–
See 'n say	17	64	3.765	●	●	●	●	–	–
Single-use camera	20	118	5.900	●	●	●	–	–	–
Stapler	16	68	4.250	●	●	–	–	–	–

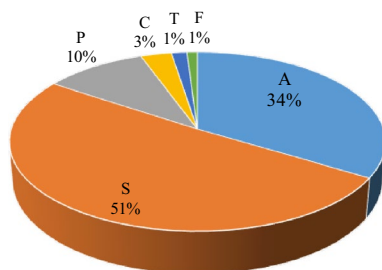


Fig. 4 Distribution of interface types in the product set

Components 2 (Upper Housing) and 3 (Lower Housing) in Fig. 6]. In this figure, the resistor C2–C3 represents the connection between C2.UH and C3.LH; so, the equivalent resistance measurement is obtained by connecting the nodes of the Resistance Measurement element to the components C2.UH and C3.LH. The Display element shows the change-resistance value between C2.UH and C3.LH, and the change-resistance value includes direct and indirect design dependencies between Upper Housing and Lower Housing.

If a positive resistance value is displayed between two components having no direct connections, then we can quickly identify that there exists an indirect connection between the two components. For example, Top Cover (Component 1) and Lower Housing (Component 3) in Fig. 6 do

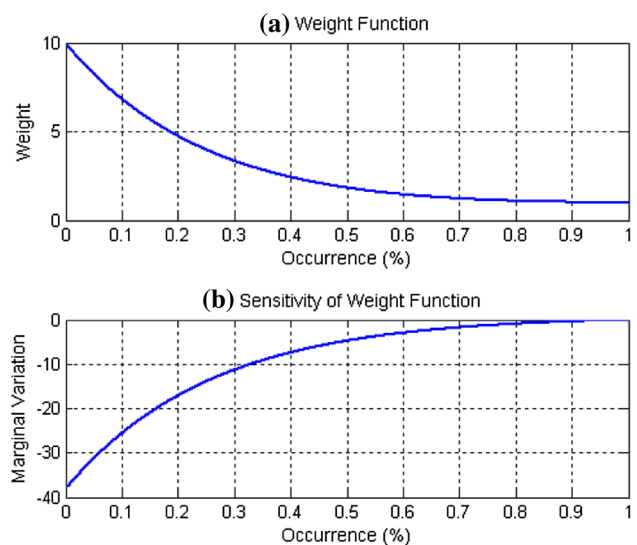


Fig. 5 Weight function and its corresponding sensitivity

not have a direct connection between them, but the change-resistance value of their connection in Fig. 9 is 0.153. Thus, this result shows that Top Cover and Lower Housing have indirect connections and corresponding design dependencies between them.

The levels of design dependencies for each interface in Fig. 9 can also be visualized in color, as shown in

Table 3 Assigned weights for different interface types

	A	S	P	C	T	F
Occurrence (%)	33.634	50.719	10.062	3.039	1.520	1.027
Weight	2.974	1.795	6.832	8.923	9.445	9.627

Table 4 Final weights for each interface type and all combinations of interfaces

Interface	Weight	Interface	Weight	Interface	Weight	Interface	Weight
A	2.974	SC	21.435	ATC	64.025	PCF	76.142
S	1.795	SF	22.843	ATF	66.137	TCP	75.598
P	6.832	SP	17.253	ATP	57.752	TFP	77.710
C	8.923	TC	36.735	ACF	64.568	ASTC	92.546
T	9.445	TF	38.143	ACP	56.184	ASTF	95.362
F	9.627	PF	32.916	AFP	58.296	ASTP	84.182
AS	9.538	CF	37.098	STC	60.488	ATCF	123.872
AT	24.838	PC	31.508	STF	62.600	ATCP	112.692
AC	23.793	AST	42.642	STP	54.215	ACFP	113.418
AF	25.201	ASC	41.074	SCF	61.032	ASTCF	163.815
AP	19.611	ASF	43.186	SCP	52.647	ASTCP	149.841
ST	22.480	ASP	34.801	SFP	54.759	ASTCFP	237.568

Fig. 6 Interfaces for the computer mouse

Mouse 1000		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1		S(3)		A,S(2)								
Upper Housing	2	S(3)		S	A(2),S								
Lower Housing	3		S		A(3),S	S	S	S	S(2)	AS	AS	AS	S(2)
PCB	4	A,S(2)	A(2),S	A(3),S		S,ST	S,F	S					
Wheel	5			S	S,ST								
Lens	6			S	S,F								
On/Off Button	7			S	S								
Battery Cover	8				S(2)								
Non-friction Strip 1	9				AS								
Non-friction Strip 2	10				AS								
Product Label	11				AS								
Battery Label	12				S(2)								

Fig. 7 wMCS values for the computer mouse

Mouse 1000		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1		5.385		6.564								
Upper Housing	2	5.385		1.795	7.743								
Lower Housing	3		1.795		10.717	1.795	1.795	1.795	3.59	9.538	9.538	9.538	3.59
PCB	4	6.564	7.743	10.717		24.275	11.422	1.795					
Wheel	5			1.795	24.275								
Lens	6			1.795	11.422								
On/Off Button	7			1.795	1.795								
Battery Cover	8				3.59								
Non-friction Strip 1	9				9.538								
Non-friction Strip 2	10				9.538								
Product Label	11				9.538								
Battery Label	12				3.59								

Fig. 8 Electric circuit model for the computer mouse

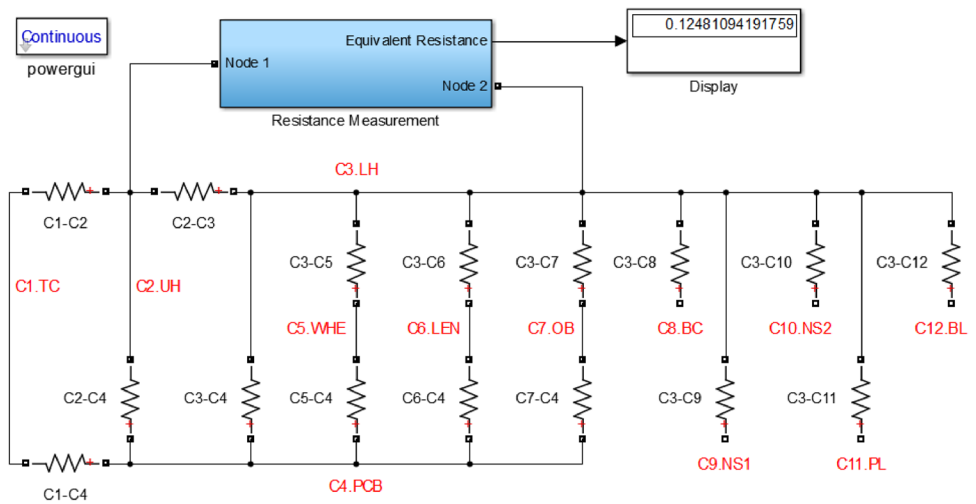


Fig. 9 Change resistances for the computer mouse

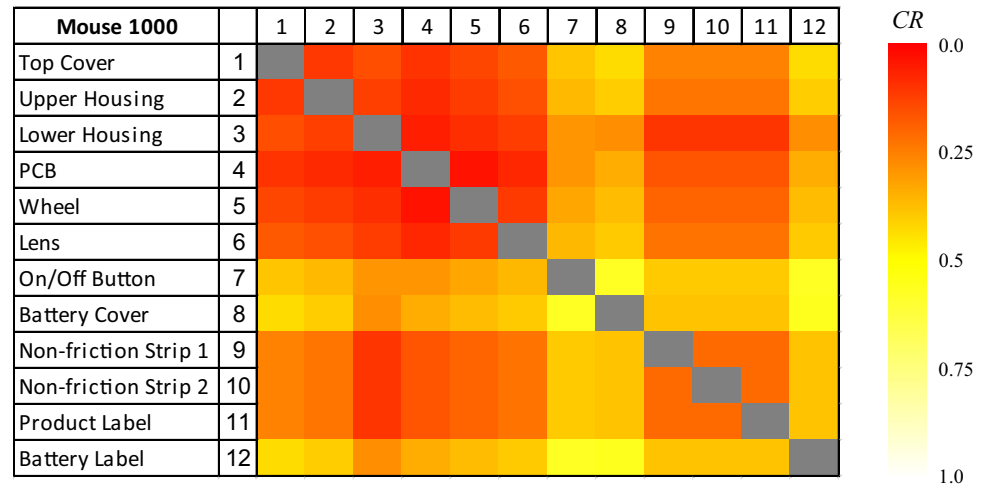
Mouse 1000		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1		0.108	0.153	0.100	0.138	0.176	0.390	0.432	0.258	0.258	0.258	0.432
Upper Housing	2	0.108		0.125	0.081	0.119	0.156	0.366	0.403	0.230	0.230	0.230	0.403
Lower Housing	3	0.153	0.125		0.061	0.091	0.121	0.294	0.279	0.105	0.105	0.105	0.279
PCB	4	0.100	0.081	0.061		0.039	0.077	0.294	0.340	0.166	0.166	0.166	0.340
Wheel	5	0.138	0.119	0.091	0.039		0.114	0.328	0.370	0.196	0.196	0.196	0.370
Lens	6	0.176	0.156	0.121	0.077	0.114		0.362	0.400	0.226	0.226	0.226	0.400
On/Off Button	7	0.390	0.366	0.294	0.294	0.328	0.362		0.572	0.399	0.399	0.399	0.572
Battery Cover	8	0.432	0.403	0.279	0.340	0.370	0.400	0.572		0.383	0.383	0.383	0.557
Non-friction Strip 1	9	0.258	0.230	0.105	0.166	0.196	0.226	0.399	0.383		0.210	0.210	0.383
Non-friction Strip 2	10	0.258	0.230	0.105	0.166	0.196	0.226	0.399	0.383	0.210		0.210	0.383
Product Label	11	0.258	0.230	0.105	0.166	0.196	0.226	0.399	0.383	0.210	0.210		0.383
Battery Label	12	0.432	0.403	0.279	0.340	0.370	0.400	0.572	0.557	0.383	0.383	0.383	

Fig. 10. Hamraz et al. (2015) utilized a similar color scale from green to red to identify the combined risk of change propagation in the component–component DSM. We can visualize direct and indirect design dependencies within a system architecture in color from white to yellow to red. Figure 10 shows an example of a DSM dependency visualization for the computer mouse. In the figure, the color in the cell is closer to red when the CR value is relatively small (i.e., high design dependency). On the other hand, the cell color is nearly white if the CR value is close to 1 (i.e., low design dependency). In this example, interestingly, even though the rate of connection per cell in the DSM is just 0.2 (i.e., total number of connections/total number of off-diagonal cells = 48/240), all components have design dependencies between them, as shown in Fig. 10. This is because most of the components are indirectly connected with the other components. In the DSM, the design dependencies for On/Off Button, Battery

Cover, and Battery Label are less than the other components, while Top Cover, Upper Housing, Lower Housing, PCB, etc. have relatively high design dependencies. Thus, the proposed method can be utilized to evaluate design dependencies in current product architectures and establish a redesign strategy focusing on interfaces between components. For example, the computer mouse can be redesigned to reduce design dependencies related to Top Cover, Upper Housing, Lower Housing, PCB, etc. to minimize total design dependency in the product architecture.

As seen in this case study, the proposed design dependency measure can capture a degree of design dependency including the effects of direct and indirect connections. The information on design dependencies is also employed to identify the most or least design-dependent components and determine more flexible architectures. In the next section, we introduce a comparison study on the design dependencies of computer mice in different generations, and the effectiveness of the proposed method is demonstrated as compared to existing methods such as 0–1 representation and MCS.

Fig. 10 Visualization of change-resistance values



5 Comparison of design dependencies for wireless computer mice

Using the assigned relative weights for each interface type in Sect. 4, we compute and compare design dependencies of wireless computer mice from Microsoft in this section. In the first comparison, the design dependencies for low-end mice (i.e., wireless computer mice released in 2010 and 2014) are selected and compared. Next, the mid-range computer mice released in 2010 and 2013 are also compared in detail. Tables 5 and 6 show each specification of the computer mice selected for comparison studies.

To demonstrate the usefulness of the proposed method, the DSMs and design dependencies obtained using 0–1 representation, MCS and the proposed method are analyzed in detail. First, Fig. 11a, b shows the DSMs for the low-end mice released in 2010 and 2014 obtained using 0–1 representation, respectively. The two computer mice have the same number of components. Compared with Wireless Mobile Mouse 1000, Wireless Mobile Mouse 1850 has a new component (i.e., Wheel Cover), while the Product Label component is excluded. In this work, we compare the average of connection values, as shown in Table 7. The average connection value in the DSM (i.e., sum of connections/total number of off-diagonal cells) is computed as follows:

$$\mu = \frac{\sum_{i=1}^N \sum_{j=1, j \neq i}^N R(i, j)}{N(N-1)}, \quad (5)$$

where N is the number of components, and $R(i, j)$ is the value of the i th row and j th column within the DSM. Interestingly, the average connection values in the two DSMs are identical, as shown in Table 7, because the total numbers of connections in the DSMs are the same. In addition, even though

Table 5 Low-end computer mice released in 2010 and 2014





Product	 Wireless mobile mouse 1000	 Wireless mobile mouse 1850
MSRP	\$14.95	\$14.95
Release date	Oct. 2010	Jun. 2014

Table 6 Mid-range computer mice released in 2010 and 2013

Product	 Wireless mobile mouse 3500	 Sculpt mobile mouse
MSRP	\$29.95	\$29.95
Release date	Jun. 2010	Aug. 2013

we also create 0–1 DSMs by taking into account indirect connections between components (i.e., in 0–1 DSMs, we can easily check the existence of indirect paths without using specific methods to compute indirect dependencies), the DSMs for the computer mice are identical, because all kinds of off-diagonal cells have “1” without any empty cells. This result shows that we cannot identify the difference of design dependencies between the two kinds of computer mice when the 0–1 representation method is employed to evaluate design dependencies. This is because the 0–1 DSMs cannot include the information on different interface types and connection strengths. Thus, the 0–1 representation method needs less information for relationships between components (i.e., information on the existence of interfaces) than other methods, but the created 0–1 DSMs do not provide refined analysis results for comparing design dependencies.

Mouse 1000		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1	■	1	1									
Upper Housing	2	1	■	1	1								
Lower Housing	3		1	■	1	1	1	1	1	1	1	1	1
PCB	4	1	1	1	■	1	1	1					
Wheel	5			1	1	■							
Lens	6			1	1		■						
On/Off Button	7			1	1			■					
Battery Cover	8			1					■				
Non-friction Strip 1	9			1						■			
Non-friction Strip 2	10			1							■		
Product Label	11			1								■	
Battery Label	12			1									■

(a) Wireless Mobile Mouse 1000

Mouse 1850		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1	■	1	1									
Upper Housing	2	1	■	1	1								
Lower Housing	3		1	■	1	1	1	1	1	1	1	1	1
PCB	4	1	1	1	■	1	1	1					
Wheel	5			1	1	■	1						
Wheel Cover	6					1	■						
Lens	7			1	1			■					
On/Off Button	8			1	1				■				
Battery Cover	9			1						■			
Non-friction Strip 1	10			1							■		
Non-friction Strip 2	11			1								■	
Battery Label	12			1									■

(b) Wireless Mobile Mouse 1850

Fig. 11 DSMs obtained using 0–1 representation

Table 7 Comparison of average connection values

Product	μ^a (sum of connection values/total number of off-diagonal cells)		
	0–1 representation	MCS	Proposed method
Wireless mobile mouse 1000	0.242 (32/132)	0.621 (82/132)	0.265 (35.021/132)
Wireless mobile mouse 1850	0.242 (32/132)	0.636 (84/132)	0.337 (44.525/132)

On the other hand, the smaller value in the proposed method means higher design dependency

^aIn 0–1 representation and MCS, smaller average value means lower design dependency

On the other hand, the DSMs obtained using MCS can have different connection strengths for each interface. Unlike the 0–1 DSMs, the results in Fig. 12 and Table 7 show that the total connection strengths and their average value for the new computer mouse were increased compared to those for the mouse released in 2010. In other words, when the MCS method is employed, the evaluated design dependencies in the DSM for the mouse released in 2014 is higher than that for the mouse released in 2010. However, MCS can evaluate design dependencies only for direct connections between components in the DSMs; so, it is necessary to apply the MCS method with other evaluation approaches that consider indirect connections.

In Figs. 9 and 14, the proposed method was applied to evaluate the design dependencies for the computer mice architectures. The interface types for each connection are defined in Figs. 6 and 13. As shown in Figs. 9 and 14, the CR values within off-diagonal cells in the DSMs are computed by utilizing the electrical circuit models. The computed CR values include direct and indirect design dependencies between components. Figures 10 and 15 visualize each CR value in color from red to white. In these figures, we observed that the number of off-diagonal cells in yellow is increased within the DSM in Fig. 15 compared to Fig. 10. This result shows that the total design dependency for the wireless computer mouse released in 2014 is reduced as

compared with the mouse released in 2010. When we compare the average design dependencies in Table 7, the average CR value is also increased from 0.265 to 0.337 (i.e., the design dependency is decreased). This is because the new component, Wheel Cover, is only directly connected with the Wheel and has relatively low design dependencies unlike the other components connected with bus-type components such as lower housing and PCB. Table 8 shows the detailed comparison in terms of components. In Table 8, the average CR value for the i th component, \overline{CR}_i is computed as follows:

$$\overline{CR}_i = \frac{\sum_{j=1, j \neq i}^N R(i, j)}{N - 1} \tag{6}$$

Compared with the computer mouse released in 2010, the average CR values for each component are also increased for the mouse released in 2014. This result clearly shows that the wireless computer mouse released in 2014 was designed by reducing design dependencies compared to the mouse released in 2010. Thus, we confirmed that the proposed method can evaluate direct and indirect design dependencies in the DSM and provide different comparison result for the computer mice unlike the 0–1 representation and MCS methods considering only direct connections between components.

Mouse 1000		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1	█	3		3								
Upper Housing	2	3	█	1	3								
Lower Housing	3		1	█	4	1	1	1	2	4	4	4	2
PCB	4	3	3	4	█	5	2	1					
Wheel	5			1	5	█							
Lens	6			1	2		█						
On/Off Button	7			1	1			█					
Battery Cover	8			2					█				
Non-friction Strip 1	9			4						█			
Non-friction Strip 2	10			4							█		
Product Label	11			4								█	
Battery Label	12			2									█

(a) Wireless Mobile Mouse 1000

Mouse 1850		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1	█	7		2								
Upper Housing	2	7	█	4	1								
Lower Housing	3		4	█	4	1		1	1	2	4	4	2
PCB	4	2	1	4	█	5		2	1				
Wheel	5			1	5	█		1					
Wheel Cover	6						1	█					
Lens	7			1	2				█				
On/Off Button	8			1	1					█			
Battery Cover	9			2							█		
Non-friction Strip 1	10			4								█	
Non-friction Strip 2	11			4									█
Battery Label	12			2									█

(b) Wireless Mobile Mouse 1850

Fig. 12 DSMs obtained using MCS

Mouse 1850		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1	█	S(7)		S(2)								
Upper Housing	2	S(7)	█	A(3),S	S								
Lower Housing	3		A(3),S	█	A(3),S	S		S	S	S(2)	AS	AS	S(2)
PCB	4	S(2)	S	A(3),S	█	S,ST		S,F	S				
Wheel	5			S	S,ST	█	S						
Wheel Cover	6					S	█						
Lens	7			S	S,F			█					
On/Off Button	8			S	S				█				
Battery Cover	9			S(2)						█			
Non-friction Strip 1	10			AS							█		
Non-friction Strip 2	11			AS								█	
Battery Label	12			S(2)									█

Fig. 13 Interfaces for Wireless Mobile Mouse 1850

Fig. 14 Change resistances for Wireless Mobile Mouse 1850

Mouse 1850		1	2	3	4	5	6	7	8	9	10	11	12
Top Cover	1	█	0.066	0.113	0.118	0.152	0.709	0.186	0.380	0.391	0.218	0.218	0.391
Upper Housing	2	0.066	█	0.070	0.093	0.126	0.683	0.159	0.346	0.349	0.175	0.175	0.349
Lower Housing	3	0.113	0.070	█	0.055	0.086	0.643	0.117	0.292	0.279	0.105	0.105	0.279
PCB	4	0.118	0.093	0.055	█	0.039	0.596	0.077	0.292	0.334	0.160	0.160	0.334
Wheel	5	0.152	0.126	0.086	0.039	█	0.557	0.114	0.327	0.365	0.191	0.191	0.365
Wheel Cover	6	0.709	0.683	0.643	0.596	0.557	█	0.671	0.884	0.922	0.748	0.748	0.922
Lens	7	0.186	0.159	0.117	0.077	0.114	0.671	█	0.362	0.396	0.222	0.222	0.396
On/Off Button	8	0.380	0.346	0.292	0.292	0.327	0.884	0.362	█	0.571	0.397	0.397	0.571
Battery Cover	9	0.391	0.349	0.279	0.334	0.365	0.922	0.396	0.571	█	0.383	0.383	0.557
Non-friction Strip 1	10	0.218	0.175	0.105	0.160	0.191	0.748	0.222	0.397	0.383	█	0.210	0.383
Non-friction Strip 2	11	0.218	0.175	0.105	0.160	0.191	0.748	0.222	0.397	0.383	0.210	█	0.383
Battery Label	12	0.391	0.349	0.279	0.334	0.365	0.922	0.396	0.571	0.557	0.383	0.383	█

Fig. 15 Visualization of change-resistance values for Wireless Mobile Mouse 1850

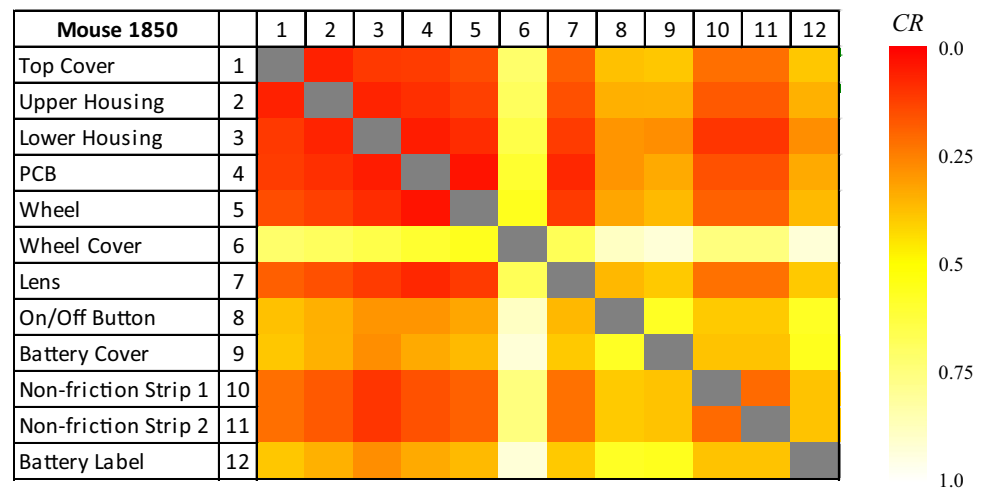


Table 8 Comparison of average CR values for each component

Components	\overline{CR}_i	
	Wireless mobile mouse 1000	Wireless mobile mouse 1850
Top cover	0.246	0.268 (↑*)
Upper housing	0.223	0.236 (↑)
Lower housing	0.156	0.195 (↑)
PCB	0.166	0.205 (↑)
Wheel	0.196	0.229 (↑)
Lens	0.226	0.266 (↑)
On/off button	0.398	0.438 (↑)
Battery cover	0.409	0.448 (↑)
Non-friction strip 1	0.251	0.290 (↑)
Non-friction strip 2	0.251	0.290 (↑)
Product label	0.251	–
Battery label	0.409	0.448 (↑)
Wheel cover	–	0.735

Mobile Mouse 1850 is increased compared to that within the wireless mobile mouse 1000

* ↑ Means that the average CR value for the corresponding component within the Wireless

In the second comparison, we analyze mid-range computer mice belonging to different generations: Wireless Mobile Mouse 3500 released in 2010 and Sculpt Mobile Mouse released in 2013. Unlike the comparison of low-end computer mice, the mid-range mice have different numbers of components, i.e., the numbers of components and interfaces for Sculpt Mobile Mouse are greater than those for Wireless Mobile Mouse 3500. In the newer computer mice, Wheel Stand, Wheel Frame, Inner Wheel, etc. were added, while Product Label and Upper stick are not included.

Figure 16 shows the DSMs created using the 0–1 representation for interfaces. Due to the increased number of components within the new computer mouse, the total number of connections between components was also increased; however, the average of connections, μ , in Eq. (5) was decreased from 0.200 to 0.137, as shown in Table 9. Meanwhile, when we employ the MCS method, the DSMs in Fig. 17 provide a similar result compared to the 0–1 DSMs. As shown in Table 9, the average of connections, μ , for the DSMs in Fig. 17 was decreased from 0.467 to 0.267. This is also because the increased rate of the total off-diagonal cells in the DSM is 150% (i.e., 240 → 600), while that of the total number of connections is just 42.9% (i.e., 112 → 160). Thus, the results analyzed by the existing methods show that the computer mouse released in 2013 has relatively low design dependencies compared to the mouse released in 2010.

Figure 19 shows the DSMs including CR values computed using the proposed method, and Table 10 summarizes the average CR values for each component within the DSMs in Fig. 19. In the table, the average CR values for all of the existing components are increased in the new DSM, and the new components such as Ball and Wheel Cover have relatively low design dependency. As shown in Table 9, the average of total design dependency for the new mouse is also lower than that for the mouse released in 2010 (i.e., higher average of CR values). This analysis result is similar to the results obtained using the 0–1 representation and MCS; however, the proposed method can provide additional information from the DSMs including direct and indirect design dependencies between components. For example, the new components such as Wheel Stand, Wheel Frame, Inner Wheel, Transceiver Slot Cover, Transceiver Slot Button, etc. have relatively high design dependencies (i.e., closer to red in color), as shown in

Table 9 Comparison of average connection values

Product	μ^a (sum of connection values/total number of off-diagonal cells)		
	0–1 representation	MCS	Proposed method
Wireless mobile mouse 3500	0.200 (28/240)	0.467 (112/240)	0.318 (76.396/240)
Sculpt mobile mouse	0.137 (82/600)	0.267 (160/600)	0.381 (228.491/600)

On the other hand, the smaller value in the proposed method means higher design dependency

^aIn 0–1 representation and MCS, smaller average value means lower design dependency

Table 10 Comparison of average CR values for each component

Components	\overline{CR}_i	
	Wireless mobile mouse 3500	Sculpt mobile mouse
Top cover	0.270	0.279 (†*)
Upper housing	0.208	0.244 (†)
Lower housing	0.177	0.210 (†)
PCB	0.187	0.218 (†)
Wheel	0.220	0.275 (†)
Lens	0.252	0.284 (†)
On/off button	0.432	0.469 (†)
Battery cover	0.437	0.477 (†)
Non-friction strip 1	0.275	0.311 (†)
Non-friction strip 2	0.275	0.311 (†)
Product label	0.275	–
Battery label	0.437	0.477 (†)
Left side cover	0.281	0.323 (†)
Right side cover	0.281	0.323 (†)
Upper stick	0.382	–
LED cover	0.707	0.752 (†)
Wheel stand	–	0.237
Wheel frame	–	0.273
Inner wheel	–	0.337
Ball	–	0.531
Wheel cover	–	0.809
Wheel frame spring	–	0.496
Transceiver slot spring	–	0.496
Transceiver slot cover	–	0.300
Transceiver slot Button	–	0.360
Windows button	–	0.331
Tail cover	–	0.398

Mobile Mouse is increased compared to that within the Wireless Mobile Mouse 3500

* † Means that the average CR value for the corresponding component within the Sculpt

Fig. 20b. Even though the total design dependency for the new mouse is relatively low, the newly added components also have more indirect connections with the other components. Therefore, we can establish additional strategies for product redesign to reduce design dependencies related to indirect connections in the product architecture. In addition, the proposed method can be used to identify the most

design-dependent component, e.g., the minimum value among the average CR values for Sculpt Mobile Mouse in Table 10 is 0.210, and its corresponding component is the Lower Housing, which has more connections than the other components.

6 Discussion and future work

This research presented a novel method to assess design dependencies within product architectures. The interfaces are classified into six different types of interfaces, and the value of each connection is computed using the wMCS formulation. To be able to evaluate the indirect design dependencies in the product, an electrical circuit analogy was proposed to model product architectures. The connections between components are considered as resistors, and the components are considered as nodes, where one or more resistors connect. Using electrical circuit models, we can measure the degree of design dependency (i.e., change resistance) including the effects of direct and indirect connections between components. In the case study, the relative weights for different types of interfaces were calculated based on the frequency of occurrence of the interface types collected from 21 electro-mechanical household products. We compared design dependencies in the DSMs for wireless computer mice obtained using 0–1 representation, MCS, and proposed method. Consequently, we observed that the proposed method provides more reliable analysis results compared to the 0–1 representation and MCS methods only considering direct connections between components.

The proposed method provides a systematic procedure to assess design dependencies within product architectures based on both direct and indirect connections. The method can be utilized either during the design stage or when benchmarking products, especially when trying to identify highly design-dependent components in products with complex interfaces. The change resistance value of a component represents the ease of change propagation. As the change resistance gets smaller for a component, design changes to this component will propagate more to connecting components. The effects of adding new features or components to an existing architecture can also be predetermined just using this approach for modeling product architectures prior to building product prototypes.

Another use of this method is as a metric to determine more flexible architectures with low design dependency. This can be a beneficial tool to be used during the new product development process when deciding between different product architectures of the same product. Depending on the strategies of the company and market trends, design engineers might have to choose between two different product prototypes to manufacture and release to market. If the product is expected to evolve quickly with developing technologies, design engineers might prefer a more flexible architecture to enable more easily modifications. The use of this method provides a simple objective manner to be able to discriminate between similar products

based on their flexibility and enables better decisions during the design process.

This method uses data extracted from a set of similar types of products. However, in some cases, designers may not have any data to assign weights for each interface type (e.g., when similar types of products do not exist in the market). Moreover, there may exist a new interface type that is not included in the proposed interface categorization. Thus, the benefits of the proposed method may be limited based on the types of products and range of interfaces being investigated. This research reflects applications and findings related to small electro-mechanical products, and further research is necessary to expand the scope and access the full potential of the proposed method. Even though similar results are expected through the investigation of other industries with more complex product architectures such as automotive or aerospace systems, investigation of a large variety of products from different size ranges might increase the accuracy of the information on interface type frequencies and relative weights.

In the proposed method, lower weights are assigned for interface types having higher frequencies of occurrence, i.e., the more frequently it is used, the more knowledge the designer has of its impact. However, an interface type may result in high design dependencies regardless of the frequency of occurrence. For example, if a new technology with new types of components and interfaces are designed into a product, then the interfaces may significantly affect the total design dependency regardless of their frequency. In this case, we would need to add more information to finalize the total design dependency of the product. For example, Min et al. (2016) consider interfaces, topological structure, and cost/benefit parameters to evaluate technology infusion impact. As a part of our future work, we plan to develop an advanced method to integrate additional information on interfaces and components into DSMs. Moreover, evaluating the economic benefits of determining design-dependent components and quantifying redesign efforts (and costs) is another possible direction for a future study. If we reflect the additional consideration in our further study, then the proposed method is expected to be more effectively used for assessing design dependencies and design-change effects of practical and complex product architectures.

Appendix: DSMs for wireless mobile mouse 3500 and sculpt mobile mouse

See Figs. 16, 17, 18, 19, and 20.

Mouse 3500		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Top Cover	1	■	1		1												
Upper Housing	2	1	■	1	1									1	1	1	
Lower Housing	3		1	■	1	1	1	1	1	1	1	1	1	1	1	1	
PCB	4	1	1	1	■	1	1	1						1	1		1
Wheel	5			1	1	■											
Lens	6			1	1		■										
On/Off Button	7			1	1			■									
Battery Cover	8			1					■								
Non-friction Strip 1	9			1						■							
Non-friction Strip 2	10			1							■						
Product Label	11			1								■					
Battery Label	12			1									■				
Left Side Cover	13		1	1	1									■			
Right Side Cover	14		1	1	1										■		
Upper Stick	15		1													■	
LED Cover	16					1											■

(a) Wireless Mobile Mouse 3500

Sculpt Mobile Mouse		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
Top Cover	1	■	1		1								1	1												
Upper Housing	2	1	■	1									1	1											1	1
Lower Housing	3		1	■	1	1	1	1	1	1	1	1	1	1		1						1	1	1		1
PCB	4	1		1	■	1	1	1							1	1	1								1	
Wheel	5			1	■												1	1	1	1						
Lens	6			1	1	■																				
On/Off Button	7			1	1		■																			
Battery Cover	8			1				■																		
Non-friction Strip 1	9			1					■																	
Non-friction Strip 2	10			1						■																
Battery Label	11			1							■															
Left Side Cover	12	1	1	1								■														
Right Side Cover	13	1	1	1									■													
LED Cover	14				1									■												
Wheel Stand	15			1	1										■	1					1					
Wheel Frame	16			1	1											1	■	1			1					
Inner Wheel	17				1												1	■	1							
Ball	18				1													1	■							
Wheel Cover	19				1														1	■						
Wheel Frame Spring	20														1	1				■						
Transceiver Slot Spring	21			1																		■	1			
Transceiver Slot Cover	22			1																			1	■	1	
Transceiver Slot Button	23			1																				1	■	
Windows Button	24		1		1																				1	■
Tail Cover	25		1	1																						1

(b) Sculpt Mobile Mouse

Fig. 16 DSMs obtained using 0–1 representation

Mouse 3500		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Top Cover	1	4			2												
Upper Housing	2	4	4	1										1	1	3	
Lower Housing	3	4	6	1	1	1	2	4	4	4	2	2	2				
PCB	4	2	1	6	5	2	1							1	1		1
Wheel	5		1	5													
Lens	6		1	2													
On/Off Button	7		1	1													
Battery Cover	8		2														
Non-friction Strip 1	9		4														
Non-friction Strip 2	10		4														
Product Label	11		4														
Battery Label	12		2														
Left Side Cover	13	1	2	1													
Right Side Cover	14	1	2	1													
Upper Stick	15	3															
LED Cover	16				1												

(a) Wireless Mobile Mouse 3500

Sculpt Mobile Mouse		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
Top Cover	1	4			2								1	1												
Upper Housing	2	4	4									1	1											4	2	
Lower Housing	3	4	4	4	1	1	2	4	4	2	2	2			4						1	3	2		1	
PCB	4	2	4	4	1	2	1								1	4	3								1	
Wheel	5		1	4													1	1	1	1						
Lens	6		1	2																						
On/Off Button	7		1	1																						
Battery Cover	8		2																							
Non-friction Strip 1	9		4																							
Non-friction Strip 2	10		4																							
Battery Label	11		2																							
Left Side Cover	12	1	1	2																						
Right Side Cover	13	1	1	2																						
LED Cover	14			1																						
Wheel Stand	15		4	4											2	2					1					
Wheel Frame	16		3	1											2	3					1					
Inner Wheel	17			1											3	1										
Ball	18			1											1	1										
Wheel Cover	19			1																						
Wheel Frame Spring	20														1	1										
Transceiver Slot Spring	21		1																							
Transceiver Slot Cover	22		3																			1	1	1		
Transceiver Slot Button	23		2																				1			
Windows Button	24	4		1																						
Tail Cover	25	2	1																							

(b) Sculpt Mobile Mouse

Fig. 17 DSMs obtained using MCS

Mouse 3500		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Top Cover	1		S(4)		S(2)												
Upper Housing	2	S(4)		A(3),S	S									S	S	S(3)	
Lower Housing	3		A(3),S		A(5),S	S	S	S	S(2)	AS	AS	AS	S(2)	A,S	A,S		
PCB	4	S(2)	S	A(5),S		S,ST	S,F	S						S	S		S
Wheel	5			S	S,ST												
Lens	6			S	S,F												
On/Off Button	7			S	S												
Battery Cover	8			S(2)													
Non-friction Strip 1	9			AS													
Non-friction Strip 2	10			AS													
Product Label	11			AS													
Battery Label	12			S(2)													
Left Side Cover	13		S	A,S	S												
Right Side Cover	14		S	A,S	S												
Upper Stick	15		S(3)														
LED Cover	16				S												

(a) Wireless Mobile Mouse 3500

Sculpt Mobile Mouse		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	
Top Cover	1		S(4)		S(2)								S	S													
Upper Housing	2	S(4)		A(3),S									S	S										S(4)	S(2)		
Lower Housing	3		A(3),S		A(3),S	S	S	S(2)	AS	AS	S(2)	A,S	A,S		A(3),S						S	A(2),S	A,S		S		
PCB	4	S(2)		A(3),S		F	S,F	S							S	A(3),S	S(3)								S		
Wheel	5			F													S	S	S	S							
Lens	6		S	S,F																							
On/Off Button	7		S	S																							
Battery Cover	8			S(2)																							
Non-friction Strip 1	9			AS																							
Non-friction Strip 2	10			AS																							
Battery Label	11			S(2)																							
Left Side Cover	12	S	S	A,S																							
Right Side Cover	13	S	S	A,S																							
LED Cover	14				S																						
Wheel Stand	15			A(3),S	A(3),S												S(2)			S							
Wheel Frame	16			S(3)	S												S(2)		A(2),S		S						
Inner Wheel	17				S													A(2),S		S							
Ball	18				S														S								
Wheel Cover	19				S																						
Wheel Frame Spring	20														S	S											
Transceiver Slot Spring	21			S																							
Transceiver Slot Cover	22			A(2),S																		S		S			
Transceiver Slot Button	23			A,S																			S		S		
Windows Button	24		S(4)	S																							
Tail Cover	25		S(2)	S																							

(b) Sculpt Mobile Mouse

Fig. 18 Interfaces in the DSMs

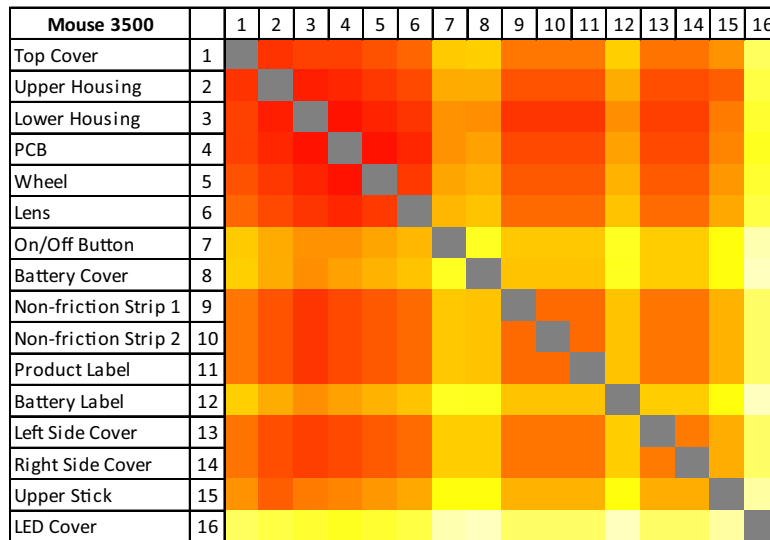
Mouse 3500		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Top Cover	1		0.101	0.128	0.127	0.163	0.198	0.396	0.407	0.233	0.233	0.233	0.407	0.226	0.226	0.287	0.684
Upper Housing	2	0.101		0.059	0.076	0.111	0.145	0.337	0.338	0.164	0.164	0.164	0.338	0.154	0.154	0.186	0.633
Lower Housing	3	0.128	0.059		0.038	0.071	0.104	0.288	0.279	0.105	0.105	0.105	0.279	0.125	0.125	0.245	0.595
PCB	4	0.127	0.076	0.038		0.039	0.076	0.288	0.316	0.143	0.143	0.143	0.316	0.142	0.142	0.262	0.557
Wheel	5	0.163	0.111	0.071	0.039		0.114	0.324	0.350	0.176	0.176	0.176	0.350	0.177	0.177	0.297	0.596
Lens	6	0.198	0.145	0.104	0.076	0.114		0.359	0.382	0.209	0.209	0.209	0.382	0.211	0.211	0.331	0.633
On/Off Button	7	0.396	0.337	0.288	0.288	0.324	0.359		0.567	0.393	0.393	0.393	0.567	0.403	0.403	0.523	0.845
Battery Cover	8	0.407	0.338	0.279	0.316	0.350	0.382	0.567		0.383	0.383	0.383	0.557	0.404	0.404	0.524	0.874
Non-friction Strip 1	9	0.233	0.164	0.105	0.143	0.176	0.209	0.393	0.383		0.210	0.210	0.383	0.230	0.230	0.350	0.700
Non-friction Strip 2	10	0.233	0.164	0.105	0.143	0.176	0.209	0.393	0.383	0.210		0.210	0.383	0.230	0.230	0.350	0.700
Product Label	11	0.233	0.164	0.105	0.143	0.176	0.209	0.393	0.383	0.210	0.210		0.383	0.230	0.230	0.350	0.700
Battery Label	12	0.407	0.338	0.279	0.316	0.350	0.382	0.567	0.557	0.383	0.383	0.383		0.404	0.404	0.524	0.874
Left Side Cover	13	0.226	0.154	0.125	0.142	0.177	0.211	0.403	0.404	0.230	0.230	0.230	0.404		0.239	0.340	0.699
Right Side Cover	14	0.226	0.154	0.125	0.142	0.177	0.211	0.403	0.404	0.230	0.230	0.230	0.404	0.239		0.340	0.699
Upper Stick	15	0.287	0.186	0.245	0.262	0.297	0.331	0.523	0.524	0.350	0.350	0.350	0.524	0.340	0.340		0.819
LED Cover	16	0.684	0.633	0.595	0.557	0.596	0.633	0.845	0.874	0.700	0.700	0.700	0.874	0.699	0.699	0.819	

(a) Wireless Mobile Mouse 3500

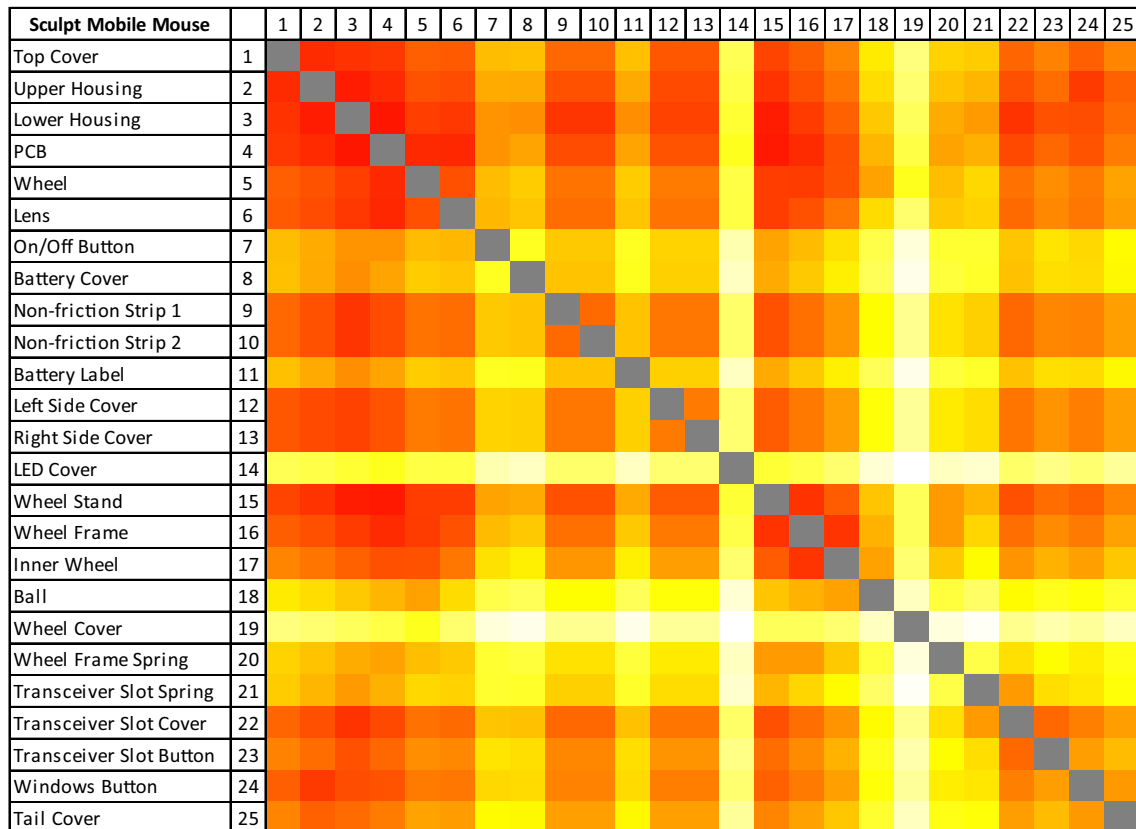
Sculpt Mobile Mouse		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Top Cover	1		0.084	0.098	0.109	0.188	0.178	0.371	0.377	0.203	0.203	0.377	0.171	0.171	0.666	0.135	0.186	0.260	0.462	0.745	0.414	0.402	0.199	0.258	0.188	0.262
Upper Housing	2	0.084		0.055	0.083	0.161	0.149	0.336	0.334	0.160	0.160	0.334	0.147	0.147	0.640	0.101	0.157	0.231	0.434	0.718	0.382	0.359	0.156	0.215	0.115	0.192
Lower Housing	3	0.098	0.055		0.044	0.122	0.109	0.290	0.279	0.105	0.105	0.279	0.130	0.130	0.601	0.055	0.116	0.191	0.394	0.679	0.339	0.304	0.101	0.160	0.151	0.210
PCB	4	0.109	0.083	0.044		0.081	0.076	0.290	0.323	0.149	0.149	0.323	0.163	0.163	0.557	0.050	0.086	0.158	0.358	0.638	0.321	0.348	0.145	0.204	0.164	0.243
Wheel	5	0.188	0.161	0.122	0.081		0.157	0.369	0.400	0.227	0.227	0.400	0.241	0.241	0.638	0.121	0.119	0.161	0.319	0.557	0.373	0.425	0.222	0.282	0.243	0.321
Lens	6	0.178	0.149	0.109	0.076	0.157		0.360	0.387	0.214	0.214	0.387	0.229	0.229	0.634	0.121	0.160	0.233	0.433	0.714	0.394	0.412	0.209	0.269	0.233	0.309
On/Off Button	7	0.371	0.336	0.290	0.290	0.369	0.360		0.568	0.394	0.394	0.568	0.414	0.414	0.847	0.320	0.368	0.442	0.644	0.926	0.598	0.593	0.390	0.449	0.425	0.494
Battery Cover	8	0.377	0.334	0.279	0.323	0.400	0.387	0.568		0.383	0.383	0.557	0.408	0.408	0.880	0.334	0.395	0.469	0.673	0.957	0.618	0.582	0.379	0.438	0.430	0.489
Non-friction Strip 1	9	0.203	0.160	0.105	0.149	0.227	0.214	0.394	0.383		0.210	0.383	0.235	0.235	0.706	0.160	0.221	0.296	0.499	0.784	0.444	0.409	0.205	0.265	0.256	0.315
Non-friction Strip 2	10	0.203	0.160	0.105	0.149	0.227	0.214	0.394	0.383	0.210		0.383	0.235	0.235	0.706	0.160	0.221	0.296	0.499	0.784	0.444	0.409	0.205	0.265	0.256	0.315
Battery Label	11	0.377	0.334	0.279	0.323	0.400	0.387	0.568	0.557	0.383	0.383		0.408	0.408	0.880	0.334	0.395	0.469	0.673	0.957	0.618	0.582	0.379	0.438	0.430	0.489
Left Side Cover	12	0.171	0.147	0.130	0.163	0.241	0.229	0.414	0.408	0.235	0.235	0.408		0.239	0.721	0.179	0.237	0.311	0.515	0.798	0.461	0.434	0.230	0.290	0.248	0.314
Right Side Cover	13	0.171	0.147	0.130	0.163	0.241	0.229	0.414	0.408	0.235	0.235	0.408	0.239		0.721	0.179	0.237	0.311	0.515	0.798	0.461	0.434	0.230	0.290	0.248	0.314
LED Cover	14	0.666	0.640	0.601	0.557	0.638	0.634	0.847	0.880	0.706	0.706	0.880	0.721	0.721		0.607	0.643	0.715	0.915	1.196	0.878	0.905	0.702	0.761	0.721	0.800
Wheel Stand	15	0.135	0.101	0.055	0.050	0.121	0.121	0.320	0.334	0.160	0.160	0.334	0.179	0.179	0.607		0.100	0.179	0.388	0.678	0.304	0.359	0.156	0.215	0.189	0.259
Wheel Frame	16	0.186	0.157	0.116	0.086	0.119	0.160	0.368	0.395	0.221	0.221	0.395	0.237	0.237	0.643	0.100		0.104	0.349	0.676	0.304	0.420	0.217	0.276	0.241	0.317
Inner Wheel	17	0.260	0.231	0.191	0.158	0.161	0.233	0.442	0.469	0.296	0.296	0.469	0.311	0.311	0.715	0.179	0.104		0.319	0.718	0.395	0.494	0.291	0.351	0.315	0.391
Ball	18	0.462	0.434	0.394	0.358	0.319	0.433	0.644	0.673	0.499	0.499	0.673	0.515	0.515	0.915	0.388	0.349	0.319		0.876	0.622	0.698	0.495	0.554	0.517	0.594
Wheel Cover	19	0.745	0.718	0.679	0.638	0.557	0.714	0.926	0.957	0.784	0.784	0.957	0.798	0.798	1.196	0.678	0.676	0.718	0.876		0.931	0.983	0.779	0.839	0.800	0.878
Wheel Frame Spring	20	0.414	0.382	0.339	0.321	0.373	0.394	0.598	0.618	0.444	0.444	0.618	0.461	0.461	0.878	0.304	0.304	0.395	0.622	0.931		0.643	0.440	0.499	0.468	0.541
Transceiver Slot Spring	21	0.402	0.359	0.304	0.348	0.425	0.412	0.593	0.582	0.409	0.409	0.582	0.434	0.434	0.905	0.359	0.420	0.494	0.698	0.983	0.643		0.304	0.436	0.455	0.514
Transceiver Slot Cover	22	0.199	0.156	0.101	0.145	0.222	0.209	0.390	0.379	0.205	0.205	0.379	0.230	0.230	0.702	0.156	0.217	0.291	0.495	0.779	0.440	0.304		0.205	0.252	0.311
Transceiver Slot Button	23	0.258	0.215	0.160	0.204	0.282	0.269	0.449	0.438	0.265	0.265	0.438	0.290	0.290	0.761	0.215	0.276	0.351	0.554	0.839	0.499	0.436	0.205		0.311	0.370
Windows Button	24	0.188	0.115	0.151	0.164	0.243	0.233	0.425	0.430	0.256	0.256	0.430	0.248	0.248	0.721	0.189	0.241	0.315	0.517	0.800	0.468	0.455	0.252	0.311		0.300
Tail Cover	25	0.262	0.192	0.210	0.243	0.321	0.309	0.494	0.489	0.315	0.315	0.489	0.314	0.314	0.800	0.259	0.317	0.391	0.594	0.878	0.541	0.514	0.311	0.370	0.300	

(b) Sculpt Mobile Mouse

Fig. 19 Change resistances



(a) Wireless Mobile Mouse 3500



(b) Sculpt Mobile Mouse

Fig. 20 Visualization of change-resistance values

References

- Ahmad N, Wynn DC, Clarkson PJ (2013) Change impact on a product and its redesign process: a tool for knowledge capture and reuse. *Res Eng Des* 24(3):219–244
- Ariyo O, Eckert CM, Clarkson PJ (2010) Towards a decentralised approach to modelling connectivity in complex products. *ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. ASME, Montreal, Quebec, Canada, pp 35–45
- Bettig B, Gershenson JK (2006) Module interface representation. *ASME Design Engineering Technical Conferences*. ASME, Philadelphia, PA
- Bozzo E, Franceschet M (2013) Resistance distance, closeness, and betweenness. *Soc Netw* 35(3):460–469
- Browning TR (2001) Applying the design structure matrix to system decomposition and integration problems: a review and new directions. *IEEE Trans Eng Manage* 48(3):292–306
- Clarkson PJ, Simons C, Eckert C (2004) Predicting change propagation in complex design. *J Mech Des* 126(5):788–797
- DeCarlo RA, Lin P-M (1995) *Linear circuit analysis: time domain, phasor, and Laplace transform approaches*. Prentice-Hall, Inc
- Dobberfuhr A, Lange MW (2009) Interfaces per module: is there an ideal number? *ASME International Design Engineering Technical Conferences- Computers and Information in Engineering Design Conference*. ASME, San Diego, CA
- Eckert C, Clarkson PJ, Zanker W (2004) Change and customisation in complex engineering domains. *Res Eng Des* 15(1):1–21
- Eppinger SD, Browning TR (2012) *Design structure matrix methods and applications*. MIT Press
- Ericsson A, Erixon G (1999) *Controlling design variants: modular product platforms*. ASME, New York
- Giffin M, de Weck O, Bounova G, Keller R, Eckert C, Clarkson PJ (2009) Change propagation analysis in complex technical systems. *J Mech Des* 131(8):081001
- Hamraz B, Clarkson PJ (2015) Industrial evaluation of FBS Linkage—a method to support engineering change management. *J Eng Des* 26(1–3):24–47
- Hamraz B, Caldwell NHM, Clarkson PJ (2012) A multidomain engineering change propagation model to support uncertainty reduction and risk management in design. *J Mech Des* 134(10):100905
- Hamraz B, Hisarciklilar O, Rahmani K, Wynn DC, Thomson V, Clarkson PJ (2013) Change prediction using interface data. *Concurr Eng* 21(2):141–154
- Hamraz B, Caldwell NHM, Ridgman TW, Clarkson PJ (2015) FBS Linkage ontology and technique to support engineering change management. *Res Eng Des* 26(1):3–35
- Hirtz J, Stone R, McAdams D, Szykman S, Wood K (2002) A functional basis for engineering design: reconciling and evolving previous efforts. *Res Eng Des* 13(2):65–82
- Holley V, Jankovic M, Yannou B (2014) Physical interface ontology for management of conflicts and risks in complex systems. *Concurr Eng* 22(2):148–161
- Hölttä KM, Otto KN (2005) Incorporating design effort complexity measures in product architectural design and assessment. *Des Stud* 26(5):463–485
- Hundal M (1990) A systematic method for developing function structures, solutions and concept variants. *Mech Mach Theory* 25(3):243–256
- Jankovic M, Holley V, Yannou B (2012) Multiple-domain design scorecards: a method for architecture generation and evaluation through interface characterisation. *J Eng Des* 23(10–11):746–766
- Jarratt T, Eckert C, Clarkson PJ (2004) Development of a product model to support engineering change management. In: *Theory and methods of competitive engineering (TMCE) 2004*. Lausanne, Switzerland, pp 331–344
- Jarratt TAW, Eckert CM, Caldwell NHM, Clarkson PJ (2011) Engineering change: an overview and perspective on the literature. *Res Eng Des* 22(2):103–124
- Keller R, Eger T, Eckert CM, Clarkson PJ (2005) Visualising change propagation. In: *The 15th International Conference on Engineering Design (ICED 05)*, Melbourne, Australia
- Klein DJ, Randić M (1993) Resistance distance. *J Math Chem* 12(1):81–95
- Ko Y-T (2013) Optimizing product architecture for complex design. *Concurr Eng* 21(2):87–102
- Koh ECY, Caldwell NHM, Clarkson PJ (2012) A method to assess the effects of engineering change propagation. *Res Eng Des* 23(4):329–351
- Lockledge JC, Salustri FA (1999) Defining the engine design process. *J Eng Des* 10(2):109–124
- Martin MV, Ishii K (2002) Design for variety: developing standardized and modularized product platform architectures. *Res Eng Des* 13(4):213–235
- MathWorks (2015) *Matlab, R2015a ed*, MathWorks, Inc, Natick
- Min G, Suh ES, Hölttä-Otto K (2016) Impact of technology infusion on system architecture complexity. *J Eng Des* 27(9):613–635
- Moullec M-L, Bouissou M, Jankovic M, Bocquet J-C, Réquillard F, Maas O, Forgeot O (2013) Toward system architecture generation and performances assessment under uncertainty using Bayesian networks. *J Mech Des* 135(4):041002
- Pahl G, Beitz W, Feldhusen J, Grote K-H (2007) *Engineering design: a systematic approach*. Springer
- Parraguez P (2015) *A networked perspective on the engineering design process: at the intersection of process and organisation architectures*. Ph.D. Dissertation, Technical University of Denmark
- Pasqual MC, de Weck OL (2012) Multilayer network model for analysis and management of change propagation. *Res Eng Des* 23(4):305–328
- Pimmler TU, Eppinger SD (1994) Integration analysis of product decompositions. In: *ASME 6th International Conference on Design Theory and Methodology*. ASME, Minneapolis, MN
- Sanchez R (1994) *Towards a science of strategic product design: system design, component modularity, and product leveraging strategies*. In: *2nd International Product Development Conference on New Approaches to Development and Engineering*, Brussels, Belgium
- Sosa ME, Eppinger SD, Rowles CM (2003) Identifying modular and integrative systems and their impact on design team interactions. *J Mech Des* 125(2):240–252
- Sosa ME, Eppinger SD, Rowles CM (2007) A network approach to define modularity of components in complex products. *J Mech Des* 129(11):1118–1129
- Stephenson K, Zelen M (1989) Rethinking centrality: methods and examples. *Soc Netw* 11(1):1–37
- Steward DV (1981) *Systems analysis and management: structure, strategy and design*. Petrocelli Books, Inc., New York
- Suh ES, de Weck OL, Chang D (2007) Flexible product platforms: framework and case study. *Res Eng Des* 18(2):67–89
- Tilstra AH, Seepersad CC, Wood KL (2009) Analysis of product flexibility for future evolution based on design guidelines and a high-definition design structure matrix. In: *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, pp 951–964

- Tilstra AH, Seepersad CC, Wood KL (2012) A high-definition design structure matrix (HDDSM) for the quantitative assessment of product architecture. *J Eng Des* 23(10–11):767–789
- Ye Y, Jankovic M, Kremer GE (2015) Understanding the impact of subjective uncertainty on architecture and supplier identification in early complex systems design. *ASCE-ASME J Risk Uncert Eng Syst Part B Mech Eng* 1(3):031005-031005-11

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Research in Engineering Design is a copyright of Springer, 2018. All Rights Reserved.