UNDERSTANDING ADAPTABILITY THROUGH LAYER DEPENDENCIES

Robert Schmidt III[1], Jason Deamer[1], Simon Austin[1]
(1) Department of Civil and Building Engineering, Loughborough University, UK

ABSTRACT

This paper looks at change from the perspective of building design (i.e. building adaptability), and how a better understanding of product architecture can bring about an easier accommodation of change for an unforeseeable future. The work explores the use of a design structure matrix (DSM) to understand the building's capacity to accommodate change using building decomposition methods (Brand's layers) and component interactions as initial guides to suggest possible product architectures. Research for this study took place along side the design stage of an ongoing BSF school project. The systematic analysis of design drawings and reports was undertaken in three phases: code documents using Brand's layers; identify all variant components to create a work breakdown structure; and classification of all component relationships populating a DSM. Simple principles, such as achieving modularity between component dependencies, can potentially reveal the implication of changing components. Insights that have been gained through the data include the appropriate layer placement of components, the possibilities of new/ different layers, and the highlighting of unwanted/ hidden dependencies. The DSM permutations have also provided a deeper understanding of the software used and its algorithmic behavior, giving greater clarity of the organization of the components, and the development of component typologies in an effort to provide a consistent, logical approach to refining the matrix.

Keywords: Adaptability, DSM, Modularization, Layers, Product Architecture, Dependencies

1 INTRODUCTION

1.1 Change

There are a large number of events that impact the performance of buildings over their lives [1]. The diverse nature of human beings and their sometimes complex needs are often a catalyst for these events, which bring about various forms of change. This paper examines change from the perspective of building design (i.e. building adaptability), and how a better understanding of the product architecture can bring about an easier accommodation of such change for an unforeseeable future. Buildings are often objects of transience - exhibiting morphological change throughout their life responding to an evolving context - not static, inflexible artifacts that are left to age and be conditioned through periodic maintenance [2, 3].The future capacity for a building to respond to changing conditions is intrinsic to many of the initial design decisions that form the product architecture [4]. However, one can design for future change in a way that reduces risk, future cost and effort; this is a growing challenge for designers, as sustainability and re-use become more critical. In the past, advocates have faced challenges of an industry focused on short-term thinking through conventional financial schemes focused on initial costs, briefs built around today's needs, and procurement routes centered on restrictive and binding contracts.

Adaptability is rarely considered in building design as a fully embodied design principle. Instead elements of adaptability are introduced periodically arising through unplanned, fragmented needs in time [5]. There is an increasing need to include adaptability as a design principle for environmental and economic reasons to provide a building fit to current and future users in a way that allows them to carry out the diverse activities required [6]. For example the pressure to recycle and conserve the earth's natural resources encourages buildings that can be reused and reconfigured to changing needs, instead of being demolished at the end of its 'usable' life, with limited recycling of components [7].

Economically, long-term operational, maintenance, and adaptation costs by far outweigh initial capital costs [8], although clients can often make a business case on first use alone, discouraging a whole-life appraisal and the designing-in of adaptability for future changes.

1.2 Context and research questions

This research is part of a multi-disciplinary study of adaptability in buildings. The live project analyzed is part of the Building Schools for the Future (BSF) scheme - a government initiative to improve schools across England – of which a key principle is to create buildings that are adaptable to change and continued future use. The purpose is to understand how building components can be modeled during design to reveal the potential ability for the product architecture to adapt to change. The analytical stage of the research focused on modeling product architecture with a Design Structure Matrix (DSM). The nested research questions were as follows. (1) Can a DSM illustrate the impact of change in a building and enhance our understanding of adaptability? If this is the case then: (2) how should components be grouped (and are Brand's layers appropriate)?; and (3) can a DSM help classify the layer/component relationships.

2 ADAPTABLE PRODUCT ARCHITECURE IN CONSTRUCTION

2.1 Designing for Adaptability (DfAD)

The expected long life of buildings, the physical scale, the number of actors and components involved, and the symbiotic relationship with its contextual surroundings conspire to make buildings complex products in a fast changing world. In the product manufacturing industry items become obsolete at such a high rate that the finished product pushed onto the market will be redesigned and improved to meet the users evolving needs, including shifts in technology and performance demands [9]. Mckee and Konell [10] give two approaches for product development, a high-risk commitment to a fixed and irrevocable product, or a tentative commitment to a malleable product that shifts from relying on market predictability to using adaptability as a key design feature. Redesigning and releasing a new model is problematic with buildings; thus taking the view that a building is a static object delivered as a finished product is not just high-risk, but potentially catastrophic.

A manufactured project under goes a design process with interconnected phases, dependant on rules and specifications as part of an evolving design. If uncontrolled design changes, derived from evolving requirements, propagate through the design and product development schedule, increased development costs are incurred and may result in failure to satisfy the user's needs [11]. This precarious condition towards change during the production stage can be extended into the usage stage of long-life products such as buildings, where the life of the building is constantly evolving, through a continual appropriation process exhibiting the characteristics displayed in product development. Lack of consideration for future change, leads to high refurbishment costs, greater user disruptions, and lost opportunities along with a greater chance of the building becoming prematurely obsolete [12, 4].

(DfX) paradigms aim to develop products that are likely to perform better in regards to X. Designing for adaptability (DfAD) looks to extend the longevity of a product by allowing it to accommodate changing circumstances [13, 4]. The definition adapted for this work is 'the capacity of a building to accommodate effectively the evolving demands of its context, thus maximizing value through life' [5]. Li et al. [6] suggest three existing approaches to developing an adaptable product: modular design, product platform, and mass customization. All three approaches include characteristics of modularity as a common denomenator. Alternatively product "Piggybacking" is "a strategy that enables renewed functionality of a technologically obsolete product through the integration or add-on of a secondary devise or component" [9]. Again, modularity is applied as a design principle to guide the design of new "piggyback" products. Thus, as Engel and Browning [4] point out, modularity can contribute to product adaptability and warrants consideration. Futhermore, the literature on adaptability often lists the interfaces between components as a critical design decision to ease future changes [8]. Here, clarifying the types, boundaries and configurations of relationships play a critical role in reducing the knock-on effect of change.

In contrast, buildings often suffer from an over-emphasis on appearance at the expense of how they come together [14]. The conventional approach not only lacks consideration of adaptability, but is at odds with the demands for greater sustainability. Paduart et al. [15] state that new construction,

maintenance and renovation of buildings contribute to 45% of European waste. This wastefulness is reflected in other industries, shown by the implementation of Waste Electrical and Electronic Equipment (WEEE) directive, requiring a target of recovery and reuse of 75% by weight of post-use home appliances and computer products [16]. Byggeth et al. [17] identify the slow process of "greening" products despite various proposals, tools and methods to serve the sustainbility-driven market. Such methods include "design for recycling" and "design for environment" which coincide with principles of DfAD.

Hence, for buildings to adapt to their evolving needs, they should be designed not only to permit reconfiguration during their life, but also accommodate reuse, modification and recycling of redundant components. Both Macozoma [18] and Guy and Ciarimboli [19] suggest that designing for disassembly (DfD) goes hand in hand with principles of adaptability. Canadian Standards [20] promotes the two strategies as integral stating that DfD bolsters the capacity for adaptability, while Graham [2] and Douglas [3] list designing for deconstruction as a key strategy for adaptability. This position is supported by Bischof and Blessing's [21] study into the flexibility and adaptability of new product design, and the importance of designing products for the whole life-cycle, including standardised interfaces, increased dimensions and capacity. It is therefore appropriate to examine how a building's capacity to change over time is affected by the organization of, and relationships between, components to increase longevity and ultimately disassembly.

2.2 Product Architecture

We now seek to define and clarify characteristics of a product architecture that enhance adaptability. Ulrich [22] defines product architecture as (1) the arrangement of *functional elements*; (2) the mapping from the *functional elements to physical components*; and (3) the specification of the *interfaces* among interacting physical components. Halman et al. [23] adds, "it is the way the components are integrated and linked together to form a coherent whole". Product architecture can thus be described as a way of structuring a product (system composition) and the interactions between (component relationships). Careful consideration of both is necessary to understand how easily changes can be made

2.2.1 System Composition

A method of decomposition can categorize building components. Hofer and Halman [24] investigated platform-based families of products as a way of standardising subsystems: "We use this hierarchy to identify architectural layers, and then use these layers for the separation of differentiation needs and commonality potential within a product family". Here, their use of layers classifies system and subsystem attributes, rather than a level of component categorization. Koh, et al. [25] used a classification system based on the component's likelihood of change, rather than attributes. Geyer [26] also proposed methods of decomposition or categorisation based on the optimisation of building components. These include primarily structural elements of the building, and decomposition based upon a functional paradigm. Geyer uses the example that of a roof has a structural function similar to the load bearing properties of a beam, as well as serving as an achitectural room, i.e. a multidisciplinary approach. The functional description of components provides insight to its associated effects due to change. However, the restriction of this model to purely structural elements ignores the architectural, mechanical and electrical elements affected by change.

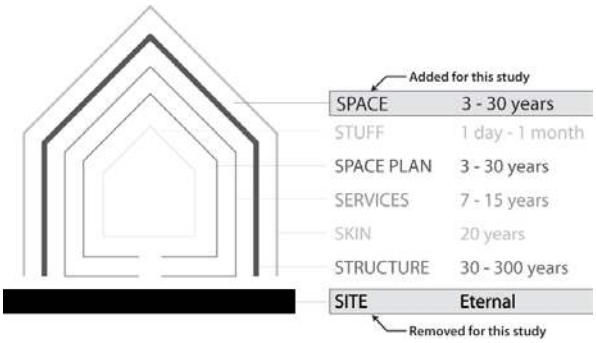

Figure 1 Brand's (1994) Layer model

Brand [27] proposes a model of building decomposition that hinges around the principle that a building is constructed from components with varying service lives, which require changing or replacing at different rates (Figure 1). As an example, Brand's model proposes that elements of the service layer (e.g. electrical, water) will change approximately every 7 to 15 years suggesting a clear separation from longer life elements such as columns or floor plates (structure, 30-300 years) and shorter life elements such as wall partitions or ceiling tiles (space plan, 3-30 years). Similar building decomposition models have been proposed

(Table 1) [28, 8, 29, 30]. The various permuations propose modifications to Brand's boundaries and nomenclature, but do not refute them.

Brand 1994	Slaughter 2001	Rush 1986	Duffy 1990	Harbarken 1998	Leupen, TBA
Site					
Structure	Structure	Structure	Shell	Skeleton	Structure
Skin	Exterior enclosure	Envelope		Infill	Skin
Services	Services	Mechanical	Services		Services
					Access
Space plan	Interior Finish Systems	Interior	Scenery		Space plan
Stuff			Set		

Table 1 Building Decomposition systems from the literature

2.2.2 Component Relationships

The relationship or dependency between components is essential in understanding how a change can be understood from a product architecture stand point. Pimmler and Eppinger [31] proposed component relationships including spatial, energy, information and materials, each type based upon *adjacency*, *flow* or *transfer*. There is no explicit consideration in this theory for the transfer of load, although material transfer would account for this in some way. Sharman et al [32] proposed additional relationships, defining their strength as well as nature. This scale ranges from high (3) to low (0) where the high rating is defined as a "Significant flow of three or more of the following; mass, energy, information, load/geometry" and low is "No significant relationship". Helmer et al [33] also proposed a rating scale to consider interfaces, including *structural*, *energy*, *signal* and *material* interactions with a range of -2 to + 2 at intervals of 0.5, where negative suggests avoidance and positive required. However, all of these have a spatial dependency, limiting its potential usefulness in construction.

Following their proposals for building decomposition, Rush [28] and Slaughter [8] also present types of component relationships (Table 2). Rush gives five categories of *remote, touching, connected, meshed,* and *unified*, thus adopting a more physical/spatial interface as opposed to flows. Slaughter defines three types of flows between components: *physical* (connection, interscetion, adjaceny), *functional* (enhance, complement, degrade) and *spatial* (independent, but interact through proximity). Century Housing System (CHS), a government lead initiative in Japan, classified relationships based on whether or not the component would be damaged once removed or changed, establishing three options: damage to both, damage to one, or not damaging [34]. In construction literature and practice the most common tactic to facilitate adaptability in this way is the choice of dry over wet connections (e.g. screws v glue or steel v concrete) allowing easier reversability (e.g., [19,18]).

Pimmler & Eppinger 1994		Rush 1986		Slaughter 2001		Helmer 2007	
Spatial	adjacency	**Remote**	do not physically touch	**Physical**	connection, intersection, adjacency	**Spatial**	adjacency
Energy	energy transfer	**Touching**	Contact w/o a permanent connection	**Functional**	enhance, complement, degrade	**Energy**	energy transfer
Information	data or signal exchange	**Connected**	permanently attached (e.g. welds)	**Spatial**	independent but in the same room	**Signal**	data exchange
Material	material exchange	**Meshed**	interpenetrate and occupy same space			**Material**	material exchange
		Unified	share physical form; no longer distinct			**Strucutral**	load exhange

Table 2 Interaction types from the literature

2.3 DSM

A Design Structure Matrix can incorporate the two principles of decomposition and dependencies in a compact visual interpretation of the modeled product architecture. Browning [35] reviews the application of DSMs in four distinct areas: Component-Based or Architecture DSM, Team-Based or Organization DSM, Activity-Based or Schedule DSM and Paramater-Based (or Low-Level Schedule) DSM. Browning outlines the simple process of a system engineering exercise of using a DSM by firstly decomposing the system into elements, understanding and documenting the interactions between elements and then analysing potential reintegration of the elements via clustering. In a static DSM, such as one modeling product architecture the goal is to cluster the elements into modules with high internal interactions and low external interactions. The designer can then quickly identify module boundaries, 'floating' components and which components comprise intra-module and external environment connections. Dependencies that exist outside the modules (i.e. between modules) highlight potential complications if one of the modules were to be removed or changed.

Whilst DSM is a well established method of analysis for research within many product fields [33], it has yet to make significant inroads into practice. Most work in the construction industry has focused

on managing the iterative design process [36] and not component-based product analysis [37].While designers in the construction industry utilize visual techinques all the time (e.g. graphic diagrams, drawings) they lack the quantitative analyis a DSM can provide. Even with the CHS project, the component-based matrices were a visual device to convey the result of a design as opposed to a design tool (e.g. clustering, sequencing) to inform possible changes [38]. For our purposes the component modules can be represented by layers of the building decomposition to investigate how components cluster within their respective layers and to ultimately inform the design process.

2.4 Summary

It can be concluded in response to question one that analysing a building's product architecture utilizing a DSM should provide novel insights into adaptability. While the above methods for system composition and component relationships provide broad, subjective guidance for a more sustainable design that seeks to avoid oblesence, they fall short of uncovering ways to reveal the impact of specific components and sub-systems on the ability to reconfigure the building later in its life. This research seeks a simple, quantifiable approach to assist designers and clients when considering future change. It builds on Schmidt III et al. [39] as a product architecture approach to explore dependencies between building components, and the way they can be decomposed into layers.

3 METHODOLOGY

The research took place along side the design stage of an ongoing BSF school project. The initial DSM model was captured through a systematic analysis of design drawings and reports submitted by the design team (architectural, structural and environmental) at the end of schematic design (RIBA Stage C). The process of abstraction was undertaken in three phases: code documents using Brand's layers; identify all variant components to create a work breakdown structure; and classify all component relationships populating a DSM. A notable limitation of the data was that it only accounted for design information up to the schematic design stage. The early nature caused some difficulty in searching for component dependencies. Due to a lack in strong definable relationships in the component specifications it was necessary to include the source and nature of the information (perceived, explicit or implied) as an ongoing record in support of future iterations (Figure 2).

A work breakdown structure (WBS) lists all of the known components that form a building. Using Microsoft Excel, components were listed in the WBS cataloguing component names, descriptions, functions, and options. Components were classified into a layer, and identified within a sub-category in each layer (e.g. foundation is a sub-category of the structure layer) – see Figure 2. Following the review of different decomposition models, Brand's layers were deemed to provide the most appropriate level of abstraction, and were the starting point for component grouping. The tabular information also allows a more comprehensive understanding of how each component may relate to others within the building and lays the foundation for exploring the appropriateness of the proposed building decomposition. A DSM was created in Microsoft Excel from the components in the WBS, establishing a matrix populated by dependencies classified as three distinct types of flows: 1. structural (e.g., gravitational, lateral), 2. spatial (e.g., adjacency, circulation), and 3. service (e.g., energy, water) – see Figure 2. These typologies build upon types proposed in the literature (e.g., physical, energy, structural) but have been translated to accommodate building terminology and change. The Excel file was then imported into Loomeo, a software that specializes in the handling of complex products. System analysis can be done in Loomeo using a matrix view that identifies groups or clusters of components based on their relationships identified by a spectral clustering algorithm [40]. The aim was to assess how the design proposals facilitate future change by analyzing the dependencies between components and manipulating their organization, with the objective of achieving highly dependent clusters in layers (modules), and minimal dependencies outside the layer (module interfaces).

4 DATA ANALYSIS

Initial trials explored the research questions to look for patterns that might inform our understanding of the hidden complexity behind the dependencies, exploring both the decomposition theory and the software. It was necessary to understand how the Loomeo algorithms worked, so any lessons drawn from the manipulations were founded on logical interpretations of component movement (i.e. is it a

manifestation of the way the software works or is it a manifestation of the problem?). Two approaches to the manipulation of the DSM were investigated. The first was manual re-sequencing of the components (Figures 3 & 4) to achieve as dense of a cluster as possible, whilst maintaining component groupings within the building layers, as established within the WBS (e.g. structure, skin). The second manipulation involved the clustering function of Loomeo (Figure 5), which sequences the components into modules based on their dependencies. In this case it is not possible to retain the grouping of components in their building layers. Our understanding of how components could shift in the matrix was two-fold – within the predetermined modules (internal shifts) and outside the predetermined modules (external shifts). Each predetermined module or layer was assigned a color easing observation of how the manipulation altered the predetermined modules (Figure 2).

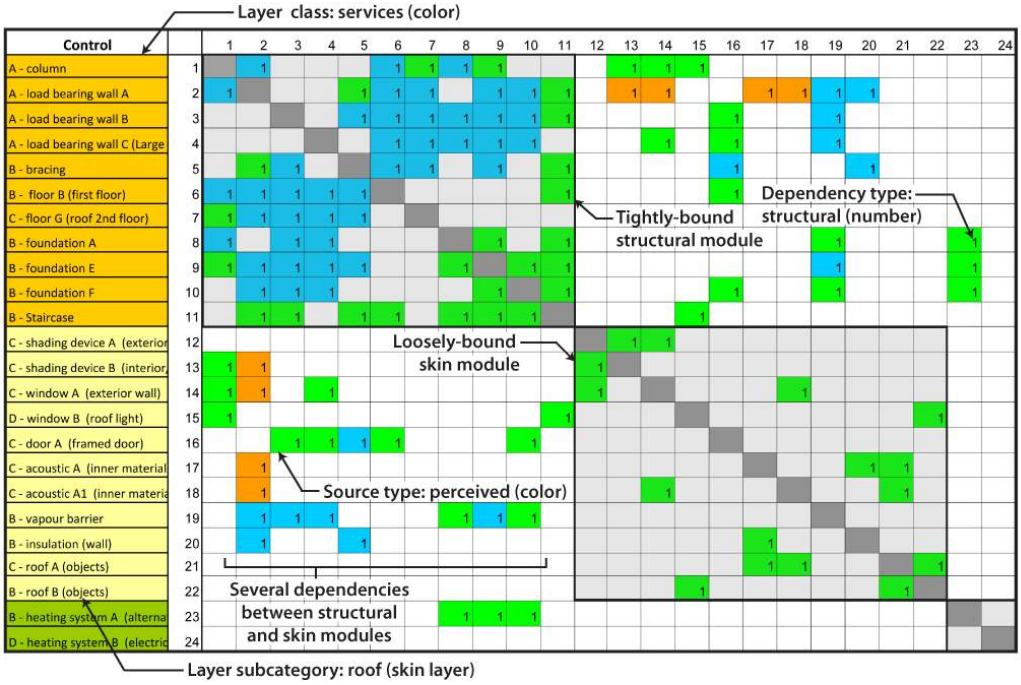

Figure 2 Part of the DSM in Excel (pre-manipulation)

4.1 Decomposition of product architecture

The first set of analyses (2 manual and 9 automatic) in line with research question two included the testing of the decomposition of product architecture into its respective layers, to test the suitability of Brand's layers, probing *how well do the components cluster in their layers?* This led to the inclusion of space as a layer in the matrix and the questioning of more, less, or different layer configurations.

4.1.1 Manual

The initial clustering process used the manual Loomeo facility to shift components to the desired location. Manual clustering allows layers to be maintained (appearing as bands of components in the DSM), which provides insights into the 'compactness' of the modules proposed by Brand (Figure 3). On the other hand, removing the boundaries of the predetermined layers allows unrestricted exploration of dependencies between components (Figure 4). A systematic process was adopted where the movement of the component was determined based on the number of dependencies it held within the layer. The objective was to position the component with the highest number of dependencies nearest to the diagonal. Components with the least dependencies were positioned towards the outside of the layer. Figure 3 shows the shuffled components inside the retained layers (space plan and services) and how certain components act as buses integrating several components inside the module. It also highlights Component 71 as an inter- and intra- module bus potentially providing a system integrating component. Inside the space plan layer, a smaller tightly-bound cluster of the sub-category acoustics can be identified indicating a potential to be classed as its own module. Figure 4 illustrates two larger tightly clustered modules with the layer restriction removed, consisting of various components mainly from the space, structure and service layers. However, some components remained more loosely-bound as all components could not be clustered tightly.

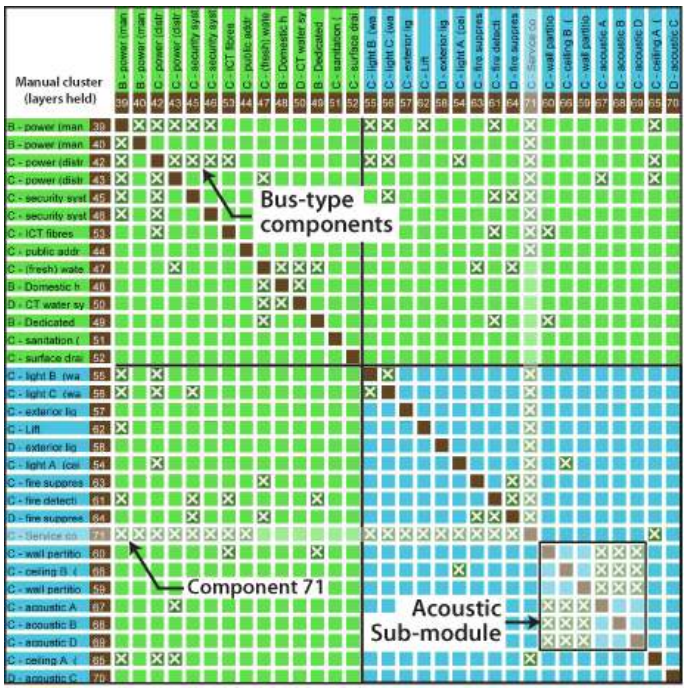

Figure 3 Manual Clustering (layers held)

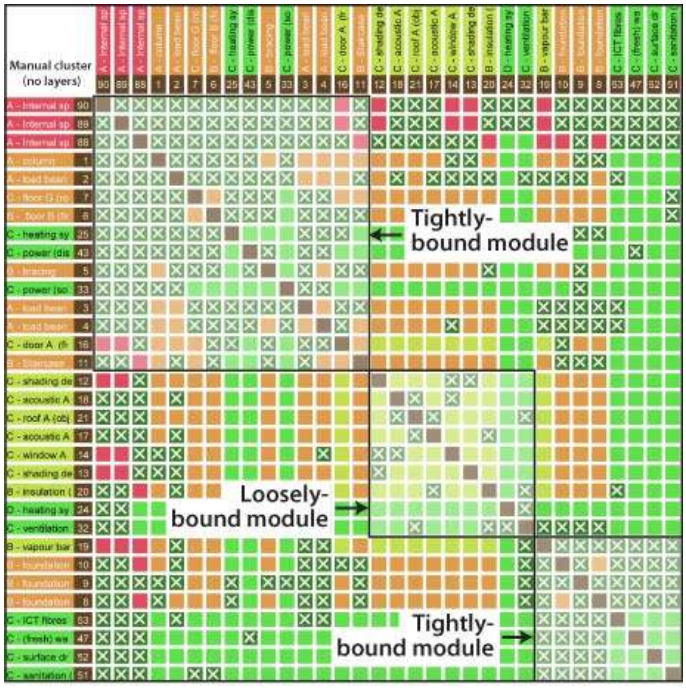

Figure 4 Manual Clustering (layers removed)

4.1.2 Automatic

Two variables were explored using Loomeo's algorithm: the cluster variable - changing the number of clusters Loomeo searches for (2-10) and the algorithmic variable – keeping the number of clusters at 6 (corresponding to the number of layers) and looking for different patterns within the iterations (10 iterations). From this, observations indicated how well components cluster. Since we had already arranged the components into predetermined modules, if the proposed solution supported the theory there would be very little movement of the components (particularly at clusters of 6). Observations of the cluster variable across two through ten clusters showed certain components repeatedly moved from their layers either to be displaced or to join into new layers. These components were often ones with little to no dependencies, while components with high dependencies tended not to move. Some displaced components showed consistent behaviour (i.e. moved to the same location), while others exhibited sporadic movement (e.g. Figure 5). Figure 5 charts the movement of components outside

their respective layers for the ten iterations of clusters at 6. The new locations of the components outside their layer were described as one of three movements: 1) outside of a cluster; 2) the extreme top/ bottom of the matrix; and 3) inside a cluster. Certain components (e.g., B79, C54) moved a high number of the times, while others (e.g., B23, C32) moved only once or twice. Components like C54 or C55 that moved 10 out of 10 times to the same location warranted initial focus.

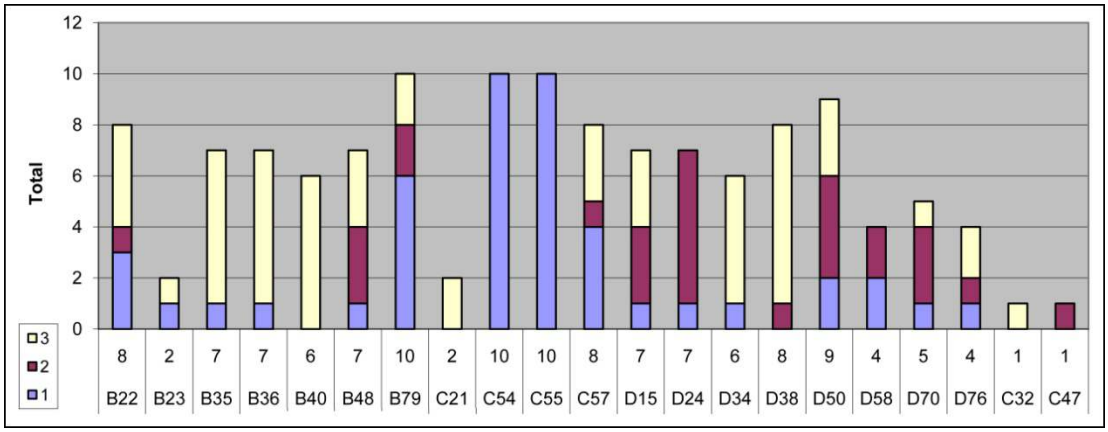

Figure 5 Components moved outside their layer

Figure 6 shows a partial DSM after automatic clustering, where the arrangement of components clearly demonstrates the breakdown of layers, with a combination of service, skin, stuff and space layers forming Module A, while Module B displays an unaffected structural layer.

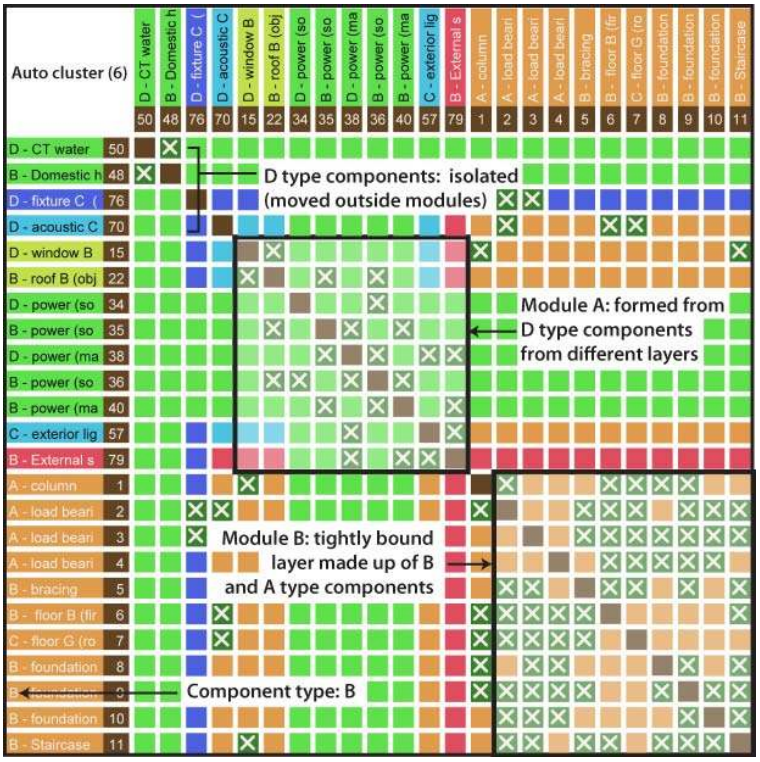

Figure 6 Part of DSM after clustering algorithm

4.2 Software trials

Following the first set of manipulations a follow-up stage (13 DSM permutations) was carried out to understand better how Loomeo operates. Issues were recorded as feedback to the software developers.

4.2.1 Influence of initial DSM structure

In the decomposition tests, it was observed that two of the layers had high dependencies (internally and externally) and did not move at all externally. This prompted the question, *if we rearrange the highly dependent components in the initial matrix, would the algorithm reunite all of the components from the predetermined layers?* As part of the test the positions of the highly dependent components

were moved from their layer and randomly scattered throughout the DSM (5 times). The algorithm was ran at a cluster size of 6, and the results showed that all the highly dependent components returned to cluster within their respective layers, implying that the algorithm had identified these components based on their highly dependent properties, not their starting position. However, the inability of Loomeo to pull any highly dependent component from its modules suggests it is unable to identify system-integrating components, i.e. those that link across several layers.

4.2.2 Separated layers

The next test was to separate the predetermined layers into smaller matrices. The software does not provide any facility to set parameters for clustering, therefore when carrying out automatic clustering it could not identify/ isolate our pre-established layers as conditional boundaries. Separation of the layers into 'new' smaller matrices was carried out into either a single module (internal reorganization) or two to three modules – the pairings looked to show the behavior of potentially highly dependent modules and more sparsely dependent ones. It was also expected that by isolating layers into individual matrices a comparison could be made with the manual clustering within layers providing possible hints towards a more optimal configuration. A further limitation of the software was the restriction of the automatic cluster function having a minimum of two clusters. This meant a single isolated layer could not be tested as a single cluster. The separated modules formed denser clusters inside their modules with less 'pull' from external dependencies (less movement outside their module), suggesting that external dependencies have an influence on the location of the components.

4.3 Component types

This section responds to research question three through the observations of automatic clustering, developing component types to express how components cluster, and establishing a classification of components that could guide more informed manual manipulation. The theory is based on the premise that the way in which a component will cluster depends on the number/ ratio of dependencies inside and outside its layer (a statistical quantification of dependencies). Four distinct types were postulated with respect to potential movement (Table 3). The theory was tested and refined by observing how the components behaved based on their type. Referring back to Figure 6, the partial DSM demonstrates how D type components formed a new cluster, while B and A type components remained forming another cluster. This behavior agrees with the expected movement for the activity types.

Component type	Movement predicted
A – highly dependent inside and outside layer	Unlikely to move, may form the core of a layer or move to extremes of matrix
B – highly dependent inside layer	No movement expected, component stays in layer towards the core
C – highly dependent outside layer	Component expected to move from layer, or would cluster near edge of layer
D – very few dependencies throughout	Movement to ends of matrix or outside of layer

Table 3 Component types and descriptions

The initial dependency-based classification was established arbitrarily, using percentage ranges. The initial ranges were < 25% for D, > 50% inside and < 50% outside a layer for B, C the reverse of this, and A > 50%. Classification of the component type was then developed using a statistical approach, according to the ratio of dependencies in a layer against those outside. Percentage ranges were calculated for each component and the set of 90 components used to establish box and whisker plots for three parameters: the ratios for inside and outside a layer and the total number of dependencies.

D component types (very few dependencies) were assessed purely on the number of dependencies as a percentage of the total number of possible dependencies for a component (90), which was calculated to be less than 11%. Component type B used a percentage range of greater than 38% population inside a layer and less than 14% outside. C was the opposite of B, with greater outside dependencies than inside the layer. A (highly dependent) was firstly determined on the total number of dependencies, similar to D, but greater than 33%. In turn, A would then become high dependencies as a B or C type.

This classification system provides a quantitative expression for highlighting potential component locations, but a single dependency may outweigh any large number of dependencies due to qualitative design issues. The classification system offers a mental 'short-cut' to the way in which components may cluster, and could potentially achieve a more consistent and rapid identification of a component's optimal location focusing on ones that do not cluster well and are potentially problematic towards future change.

5 DISCUSSION

The literature review and data analysis has provided considerable insights into the first research question. A DSM has the capacity to compactly model a building's product architecture, hence illustrating how well a proposed design can respond to change, through the clustering of modules and observing of dependency relationships in and outside a module. From a design perspective, this can suggest alternative modules (layers) or changes in the design of components through the manipulation of component locations and/ or the highlighting of unwanted/ hidden dependencies. The matrix can also visualize which components are the most appropriate to serve as intra-system interfaces or system integrators and which modules are the most applicable as product platforms (e.g. dependencies across several modules). The above provides valuable information for the designer working within an iterative process. This is supported by linking the DSM data to the source information to verify dependencies and engage the design team not only through the potentially unfamiliar matrix but also their own production drawings. Being involved in the process early allows a range of solutions to be visualized by the designer, helping them consider how the building's components interact and thereby negotiating more informed trade-offs. It is important to consider the type of dependencies that lie outside a layer (spatial, structural, service), as they influence change differently. Structural dependencies that lie outside a module may have a greater implication than spatial dependencies because of their physical connection. Each dependency type will hold a particular relevance to the type of module formed (or layer being observed).

With regards to the second question about how components should be grouped, the decomposition permutations resulted in components frequently being displaced, often leaving their layer. This observation either challenges the suitability of Brand's layers or reflects components that have not been designed appropriately for adaptability. One important pattern identified was the tracking of certain components that clustered regularly into alternative layers suggesting the need to investigate their layer placement and designed dependencies. In other cases, components clustered regularly into a newly formed layer, while others consistently formed a sub-cluster within their layer – suggesting the possibilities of additional layers. Furthermore, the identification of several unwanted dependencies (intra-module) reiterates the question of appropriate decomposition or poor design. A more detailed analysis could be undertaken of component dependencies at later stages when more detailed design data is available, providing a more definitive analysis. Additionally, further testing could be carried out comparing a greater number of decomposition models (e.g. Table 1) and system levels (i.e. the sub-categorization). Ultimately the answer may lie in further exploration of the component's functionality or design characteristics.

Due to the nested nature of the research questions, the third has been partially answered through the investigations into the first two questions. Again, the DSM has provided initial evidence into a logical and more robust system of understanding the optimal layer for a component through the development of the four component typologies (based on the number and ratio of dependencies in and outside the modules to the total number of components). This characterization, whilst accurately reflecting the behavior of some components provides hints towards the proper placement of more sporadic components. Further development of the component types will provide refined guidance for a quicker and improved means of identifying components that require further design.

6 CONCLUSION

A DSM has the potential to visualize the complete arrangement of building components and their relationships - unlike traditional design methods based around drawings and accompanying design calculations - providing a compact and powerful device. With respect to our first research question, a DSM can provide feedback on the adaptability of a proposed solution by mapping the evolving component dependencies through each design stage. Simple principles, such as achieving modularity between component dependencies, can potentially reveal the implication of changing components. Insights into question two have been gained through the data, including the appropriate layer placement of components, the possibilities of new/ different layers, and the highlighting of unwanted/ hidden dependencies. The DSM permutations have also provided a deeper understanding of the software and its algorithmic behavior giving greater clarity of the organization of the components. These benefits served as a foundation for a response to the third research question and development of the component typologies in an effort to provide a consistent, logical approach to refining the matrix.

The study has some clear limitations: it is based on a single case study, at one stage of the design process. The number of analyses was not exhaustive but clear patterns of behavior did occur. The metrics and component typologies are therefore in an early stage of development. Further developments should include the comparison of multiple building projects and analysis carried through several design stages. The goal would be to develop a design tool that incorporates lessons distilled from the DSM analysis as an integral part of the design process. Strategies and guidelines for adaptability could be used as additional guidance for identifying optimal modules and to associate design tasks or change scenarios to the static DSM as well. At each design stage an analysis of the DSM could be made (linking varying levels of abstraction to different points in the design process), where observations guided by the principles being developed feed in to the next stage of the design process and are coordinated with stakeholder roles to create refined modules (e.g. fewer dependencies outside their layer), hence creating a more adaptable solution and accommodating the potential for component reuse and recycling.

ACKNOWLEDGEMENTS

This research project is funded by the EPSRC through the IMCRC at Loughborough University. The authors also thank the participating organization involved in the case study.

REFERENCES

[1] Tschumi, B. 1996. *Architecture and disjunction.* Cambridge: MIT Press.
[2] Graham, P. 2005. *Design for adaptability - an introduction to the principles and basic strategies.* Australia: The Royal Australian Institute of Architects, GEN66.
[3] Douglas, J. 2006. *Building adaptation*. 2nd ed. Great Britian: Elsevier Ltd.
[4] Engel, A., and T. R. Browning. 2008. Designing systems for adaptability by means of architecture options. *Systems Engineering* 11 (3): 125.
[5] Schmidt III, R., Eguchi, T., Austin, S. and Gibb, A., 2010b. What is the meaning of Adaptability in the building industry?, *16th International Conference on "Open and Sustainable Building",* May 17-19 2010, Bilbao, Spain.
[6] Li, Y., Xue, D., Gu, P. (2008). Design for Product Adaptability. *Concurrent Engineering* 16, (3): 220.
[7] Gibb, A., Austin, S., Dainty, A., Davison, N., & Pasquire, C. (2007). Towards Adaptable Buildings: pre-configuration and re-configuration - two case studies. *ManuBuild 1st International Conference* . Rotterdam: Loughborough University.
[8] Slaughter, E. S. (2001). Design strategies to increase building flexibility. *Building Research & Information 29* , (3): 208.
[9] Rai, R., & Terpenny, J. (2007). Principles For Managing Technological Product Obsolescence. *International Conference On Engineering Design, ICED'07.* Paris: NSF Center for e-Design.
[10] Mckee, D., Konell, S. 1993. Product adaptability: Assessment and strategy. *Journal of Product and Brand Management 2*, (2): 33.
[11] Peterson, C., Paasch, R. K., Ge, P., & Dietterich, T. G. (2007). Product Innovation For Interdisciplinary Design Under Changing Requirements. *International Conference On Engineering Design, ICED'07.* Paris: Oregon State University.
[12] Schneider, T., Till, J. 2006. Flexible Housing: opportunities and limits. *Architectural Research Quarterly 9* (2): 157.
[13] Kasarda, M., J. Terpenny, D. Inman, K. Precoda, J. Jelesko, A. Sahin, and J. Park. 2007. Design for adaptability (DFAD) - a new concept for achieving sustainable design. *Robotics and Computer-Integrated Manufacturing* 23 : 728.
[14] Kieran, S., and J. Timberlake. 2004. Refabricating architecture: How manufacturing methodologies are poised to transform building construction. New York: McGraw-Hill.
2008.
[15] Paduart, A., Debacker, W., Henrotay, C., Temmerman, N. D., Wilde, W. P., & Hendtrickx, H. (2009). Transforming Cities: Introducing Adaptability in Existing Residential Buildings through Reuse and Disassembly Strategies for Retrofitting. *Lifecycle Design of Buildings, Systems and Materials* (pp. 18-23). Enschede: Construction Materials Stewardship.

[16] Pandey, V., Thurston, D., Kanjani, K., & Welch, J. 2007. Distributed Data Sources For Lifecycle Design. *International Conference On Engineering Design, ICED'07.* Paris: University of Illinois.
[17] Byggeth, S. H., Ny, H., Wall, J., Broman, G., & Robèrt, K.-H. R. (2007). Introductory Procedure For Sustainability-Driven Design Optimization. *International Conference On Engineering Design, ICED '07.* Paris: Blekinge Institute of Technology.
[18] Macozoma, D. 2002. Understanding the Concept of Flexibility in Design for Deconstruction, Proceedings of the CIB Task Group 39 – Deconstruction Meeting, CIB Publication 272.
[19] Guy, B., and N. Ciarimboli. 2008. Design for disassembly in the built environment: A guide to a closed-loop design and building.
[20] CSA. 2006. Guideline for design for disassembly and adaptability in buildings. Ontario: Canadian Standards Association, Z782-06.
[21] Bischof, A., & Blessing, L. 2007. Design For Flexibility: Making Provisions For Requirement Changes. *International Conference On Engineering Design, ICED'07.* Paris: University of Technology Berlin.
[22] Ulrich, K. 1995. Product architecture in the manufacturing firm. *Research Policy 24*: 419.
[23] Halman, J., Hofer, A., Vuuren, W. 2003. Platform-Driven Development of Product Families: Linking Theory with Practice. *Product Innovation Management*, 20: 149.
[24] Hofer, A. P., & Halman, J. I. (2004). Complex products and systems: Potential from using layout platforms. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2004*, 55.
[25] Koh, E. C., Keller, R., Eckert, C. M., & Clarkson, P. J. (2007). Component Classification: A Change Perspective. *9TH International Design Structure Matrix Conference, DSM'07.* Munich: Engineering Design Centre, University of Cambridge.
[26] Geyer, P. (2009). Component-oriented decomposition for multidisciplinary design optimization. *Advanced Engineering Informatics* , 12-31.
[27] Brand, S. (1994). How buildings learn: What happens after they're built. New York: Penguin.
[28] Rush, R. D. (1986). *The building systems integration handbook.* New York: John Wiley & Sons.
[29] Leupen, B., R. Heijine, and J. V. Zwol, eds. 2005. *Time-based architecture*. Rotterdam: 010 Publishers.
[30] Duffy, F. (1990). Measuring building performance. *Facilities 8, (5)* , 17.
[31] Pimmler, T. U., & Eppinger, S. D. (1994). Integration analysis of product decomposition. *ASME 6th International Conference on Design Theory and Methodology.*
[32] Sharman, D. M., Yassine, A. A., & Carlile, P. (2002). Characterising Modular Architecture. *ASME 2002 International Design Engineering Technical Conferences.* Montreal: Massachusetts Institute of Technology.
[33] Helmer, R., Yassine, A., & Meier, C. (2007). Module and Interface Identification– A Comprehensive Approach Using DSM. *9th International Design Structural Matrix Conference, DSM'07.* Munich: Technische Universität München.
[34] Century Housing Promotion Committee (CHPC). 1988. *Century housing system Pamphlet.* Japan: Center for Better Living.
[35] Browning, T. R. (2001). Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions. *IEEE Transactions on Engineering Management Vol. 48 No.3* , 292-305.
[36] Baldwin, A. N., Austin, S. A. and Waskett, P. (2009) 'Process Modelling For Planning, Managing And Control Of Collaborative Design' in *Collaborative Construction Information Management*, Shen G Q et al (Eds), Spon Press, Taylor and Francis, London, 68-79.
[37] Pektas, S.T. and Pultar, M., 2006. Modelling detailed information flows in building design with the parameter-based design structure matrix. *Design Studies,* **27**, pp. 99.
[38] Schmidt III, R., Eguchi, T. and Austin, S., 2010a. Lessons From Japan: A look at Century Housing System, *12th International Dependency and Structure Modelling Conference,* 22-23 July 2010, Cambridge, UK.
[39] Schmidt III, R., Austin, S., & Brown, D. (2009). Designing Adaptable Buildings. *11th International Design Structure Matrix Conference, DSM'09.* Greenville: Department of Civil and Building Engineering, Loughborough University.
[40] Luxburg, U. 2007. A Tutorial on Spectral Clustering. *Statistics and Computing*, 17 (4).