

Interface Structure Matrix for Analysis of Products and Processes

Andrew Kusiak

Intelligent Systems Laboratory, Mechanical and Industrial Engineering, 3131 Seamans Center, The University of Iowa, Iowa City, IA 52242-1527, USA

Abstract

The product life-cycle management has been considered from different perspectives, including the Design (or Dependency) Structure Matrix (DSM) approach. In this paper, a matrix decomposition approach is presented as an alternative to the DSM approach. The decomposition approach calls for another representation of the input data (a matrix) referred to in the literature as the Interface Structure Matrix (ISM). The ISM representation offers numerous advantages that are discussed in this paper. The Meta-ISM algorithm presented in this paper offers a new perspective for analyzing products and processes. The concepts introduced in the paper are illustrated with examples.

Keywords:

Design Structure Matrix, Interface Structure Matrix, Process Analysis Algorithm; Life Cycle Engineering

1 INTRODUCTION

The concept of the Design (or Dependency) Structure Matrix (DSM) was originally proposed in the 1960s and highlighted by Steward's publications in the 1980s (e.g., Steward 1981). The DSM concept remained essentially unnoticed until 1990. The renaissance of the DSM some fifteen years ago can be attributed to the growing interest in the theory of design, and it has now proven to be a significant and lasting contribution to engineering design.

The DSM has been studied by numerous researchers. Browning (2001) reviewed applications of different types of DSMs. Chen et al. (2003) applied the DSM concept to manage new product development projects. The schedule of the interrelated tasks was greatly improved, and as a result the impact of the delayed tasks could be better analyzed. Chen and Lin (2003) developed a numerical DSM that quantified dependencies and their coupling strength. Bilalis et al. (2004) applied the DSM concept to represent innovation-related attributes arising from the product, process, and management perspective. Cho and Eppinger (2005) presented a process model for analyzing design projects that was based on the DSM and which considered the stochastic nature and resource constraints. Xu et al. (2005) applied the DSM to a multidisciplinary cooperative design. Karniel et al. (2005) used the concept of the DSM (constructed from the geometry constraints) to decompose a complex 3D-surface-fitting reengineering problem into sub-problems. The sub-problems were optimally solved in a sequence generated from the DSM. Şule et al. (2006) applied the parameter-based DSM in construction projects to acquire a better understanding of the design process. Numerous DSM applications are surveyed in Lindemann et al. (2007).

DSM is an $n \times n$ (square) matrix, where n is the number of components (e.g., parts in a product, tasks of a project, departments within an organization) to be modelled. Each entry of the DSM represents a particular kind of relationship in

the system (e.g., belonging to the same set, information flow, force flow) between two entities (e.g., components, subassemblies, activities). For example, two parts belonging to the same assembly form an assembly relationship; two activities with information flow form an information flow dependency. Representing the complex dependencies of a system with a DSM allows the system to be decomposed into manageable subsystems. Valuable insights from the system structure can be derived after examining the interactions within and among the subsystems. An example of the DSM with nine components, labelled 1 through 9, is shown in Fig. 1. The numbers on the diagonal indicate the corresponding component number.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | | | | | | | | |
| 2 | | 2 | | | | | * | | |
| 3 | | | 3 | | * | | | | |
| 4 | | | | 4 | | | * | | |
| 5 | | | * | | 5 | | | | * |
| 6 | | * | | | | 6 | | | |
| 7 | | | | * | * | | 7 | | |
| 8 | | * | | | | * | | 8 | |
| 9 | | | | | | | | | 9 |

Figure 1: An example of the dependency structure matrix.

To build the matrix in Fig. 1, the relationships among nine components needed to be specified. In fact, the DSM allows capturing bidirectional relationships. For example, the asterisk in row 7 and column 4 implies that component (column) 4 impacts component (row) 7. Similarly, the entry (4, 7) implies that component 7 impacts component 4, i.e., the relationship between components 4 and 7 is bidirectional. The interaction medium could be information, signal, force, and so on.

DSM matrices can be established for parts, sub-assemblies, assemblies, and process activities across various technology

areas, e.g., electronics, mechanical engineering, and software.

It would appear that the matrix in Fig. 1 does not contain any additional information besides the interactions. However, transforming this matrix with the algorithm presented in Kusiak et al. (1994) produces the organized matrix in Fig. 2.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 9 | 3 | 5 | 7 | 4 | 2 | 6 | 8 |
| 1 | 1 | | | | | | | | |
| 9 | | 9 | | | | | | | |
| 3 | | | 3 | * | | | | | |
| 5 | * | * | | 5 | | | | | |
| 7 | | | | | 7 | * | | | |
| 4 | | | | | | * | 4 | | |
| 2 | | | | | * | | | 2 | |
| 6 | | | | | | | * | | 6 |
| 8 | | | | | | | * | * | 8 |

Figure 2. The organized DSM matrix of Fig. 1.

It can be easily observed in the matrix of Fig. 2 that the components 1 and 9 are independent (no asterisk is present at the entry (1, 9) and (9, 1)). Components 3 and 5, as well as 4 and 7, form cycles. Components 2, 6, and 8 exhibit a one-directional dependence. The patterns displayed in Fig. 1 can serve various purposes, e.g., modularity analysis and scheduling design activities.

2 WHAT IS THE INTERFACE STRUCTURE MATRIX?

To construct a DSM, relationships (dependencies) among the components or activities should be established. Acquiring such information from products or processes involving thousands of components or activities is difficult, if feasible at all. The relationships among components are usually obtained from two sources, geometry files (e.g., ProE) or enterprise resource planning (ERP) systems (e.g., SAP, EPICOR solutions). Most of computer-aided design (CAD) systems have been developed with a focus on expressing geometry rather than storing and accessing the files representing it. Extracting any information about relationships among components from a typical CAD system is cumbersome, if not impossible, due to the lack of suitable protocols and standards. Therefore, the process of extracting the dependencies among components from geometry files is usually performed manually. For example, design experts were interviewed to extract the dependencies for a commercial aircraft engine (Sosa et al. 2000). The DSM research typically assumes that the dependency information is available, but this assumption rarely holds in practice. As a consequence, the need to automatically generate such dependency information is apparent.

To mitigate the difficulties of generating large-scale DSMs, the use of the Interface Structure Matrix (ISM) is proposed. The ISM represents the component-interface (or activity-interface) relationships rather than the component-component (or activity-activity) relationships of the DSM. These component-interface (or activity-interface) relationships provide another perspective of the dependency relations. Processing the ISM information (e.g., clustering) leads to clusters (modules) similar to those produced from a DSM. In many cases, it is far easier and practical to obtain the interface information and construct an ISM in place of a DSM.

In addition, large DSMs are difficult to visualize, while ISMs are easy to analyze, including visual assessment.

The idea of considering interfaces in modular design is not new (Jose and Tollenaere 2005). Chen and Liu (2005) examined the impact of interfaces on product development strategies. Once the interfaces have been established, product structure is determined. Parallel design efforts on different modules are possible, significantly reducing the cycle time of the design process.

In the next two sections, the comparison between DSM and ISM concepts is discussed using process models.

2.1 Analysis of Process Models with the Dependency Structure Matrix

Consider a process model with twelve activities a through m (represented by rectangles) and dependencies (represented by arrows) in Fig. 3. A particular dependency, e.g., 5, is an output of activity d and an input to activity f. It is worth mentioning that the model in Fig. 3 follows the IDEF3 notation where the information (dependency) flows from left to right, and the top down arrow denotes controls. The IDEF3 distinction between the inputs/outputs and controls is key in process modeling, in particular product life-cycle management. Controls are often confused with the information (output/input) flow thus leading to unnecessary delays of activities. An activity transforms input into output, while the control signal triggers the corresponding activity.

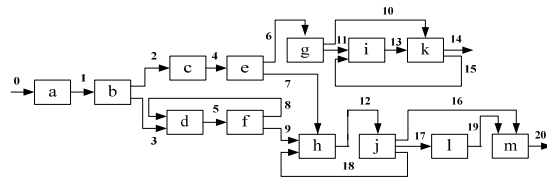


Figure 3. Process model with twelve activities labeled a - m.

The process model of Fig. 3 could be analyzed with the DSM and ISM approaches. The relationships between inputs and outputs of the activities a through m in Fig. 3 are represented as the DSM matrix in Fig. 4. Note that the output-control relationships are not considered in the matrix, as they usually occur at time scales different than the processing of inputs into outputs.

| | | Activity Output | | | | | | | | | | | | |
|----------------|---|-----------------|---|---|---|---|---|---|---|---|---|---|---|---|
| | | a | b | c | d | e | f | g | h | i | j | k | l | m |
| Activity Input | a | + | | | | | | | | | | | | |
| | b | * | + | | | | | | | | | | | |
| | c | | * | + | | | | | | | | | | |
| | d | | * | | + | * | | | | | | | | |
| | e | | | * | | + | | | | | | | | |
| | f | | | | * | | + | | | | | | | |
| | g | | | | | * | | + | | | | | | |
| | h | | | | | | * | | + | * | | | | |
| | i | | | | | | | * | | + | * | | | |
| | j | | | | | | | | * | | + | | | |
| | k | | | | | | | | | * | | + | | |
| | l | | | | | | | | | | * | | + | |
| | m | | | | | | | | | | | * | | + |

Figure 4. DSM of the process model in Fig 3.

The block-diagonal pattern shown in Fig. 5 is generated by applying the triangularization algorithm (Kusiak et al. 1994) to the DSM of Fig. 4. Four groups of activities (potential modules) M1, M2, M3, and M4 are visible in Fig. 5. The interactions between the modules are indicated by the asterisks outside of the shaded boxes in Fig. 5.

| | | Activity Output | | | | | | | | | | | | | | | | |
|----------------|---|-----------------|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|---|
| | | m | g | a | j | i | k | b | c | d | f | e | k | h | | | | |
| Activity Input | m | + | | | | | | | | | | | | | | | | |
| | g | | + | | | | | | | | | | | | | | | |
| | a | | | + | | | | | | | | | | | | | | |
| | j | | | | + | | | | | | | | | | | | | |
| | i | | * | | | + | * | | | | | | | | | | | |
| | k | | | | | | + | * | | | | | | | | | | |
| | b | | | * | | | | + | | | | | | | | | | |
| | c | | | | | | * | + | * | | | | | | | | | |
| | d | | | | | | | | * | + | * | | | | | | | |
| | f | | | | | | | | | * | + | * | | | | | | |
| | e | | | | | * | | | * | | | + | | | | | | |
| | l | | | * | | | | | | | | | + | | | | | |
| | h | | | * | | | | | | | | * | | + | | | | + |

Figure 5. The organized DSM of Fig. 4.

The graphical representation of the organized DSM in Fig. 5 is shown in Fig. 6. Note that the process in Fig. 6 includes three independent sub-processes S1, S2, and S3. Analogous to the structure in Fig. 5, the activities in Fig. 6 are arranged at four levels.

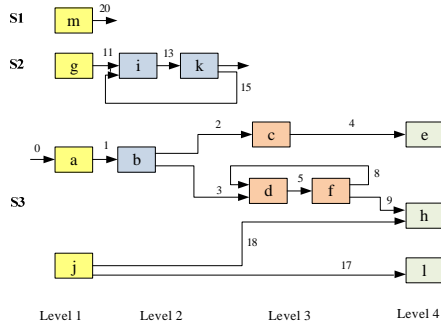


Figure 6. The organized process model of Fig. 3.

Suppose that the process in Fig. 6 needs to be redefined by removing or inserting additional activities. Performing such an operation in the DSM is quite challenging, as rows and columns are affected and all interactions need to be carefully considered. It will be shown later that such an operation is much more transparent and easier to perform in the ISM.

2.2 Process Analysis with the Interface Structure Matrix

An interface is defined here as an interaction between a component with any other object (in particular a component) in the same group. The matrix is usually not square. The ISM has the following characteristics:

- The entries of an ISM are usually available in industrial databases (e.g., from the bill of materials), and therefore the ISM can be automatically constructed with simple data transformation and extraction codes.

- Algorithms for analysis of ISMs are readily available, which is a great benefit. Data-mining algorithms (e.g., association-rule algorithms), clustering algorithms (e.g., the cluster-identification algorithm, Kusiak and Chow 1987), and mathematical-programming algorithms can be used. From the data mining perspective, an ISM is an object-feature data set where the objects (rows) are the components and the features (columns) are the interfaces. Each feature could be also considered as an interaction type. In mathematical programming models applied to ISMs, a variety of objectives and constraints can be incorporated. The models themselves can be solved with widely accessible commercial and research software. In contrast to the ISM algorithms, the algorithms for transforming DSM matrices are scarce.
- ISMs are easy to update, interpret, and observe changes. Most operations on ISMs amount to adding or removing rows and the corresponding columns, which is rather intuitive.

To illustrate the ISM, consider the process of Fig. 3 represented in its entirety, i.e., all activities, inputs, outputs, and controls, in Fig. 7. The easiest way to construct the matrix in Fig. 7 is by assigning values (I-input, O-output, and C-control) to all interfaces associated with each activity. For example, for activity h (row h in Fig. 7), the value C is assigned to interface 7, I to interface 9, O to interface 12, and I to interface 18.

| | | Interface (I, O, C) | | | | | | | | | | | | | | | | | | | | | |
|----------|---|---------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | |
| Activity | a | I | O | | | | | | | | | | | | | | | | | | | | |
| | b | | I | O | O | | | | | | | | | | | | | | | | | | |
| | c | | | I | O | | | | | | | | | | | | | | | | | | |
| | d | | | | I | O | | I | | | | | | | | | | | | | | | |
| | e | | | | | I | O | O | | | | | | | | | | | | | | | |
| | f | | | | | | I | O | O | | | | | | | | | | | | | | |
| | g | | | | | | | C | | O | O | | | | | | | | | | | | |
| | h | | | | | | | C | | I | O | | | | | | | | | I | | | |
| | l | | | | | | | | | I | O | | | | | | | | | | | | |
| | j | | | | | | | | | | C | | | | | | | O | O | O | O | | |
| | k | | | | | | | | | C | | I | O | O | | | | | | | | | |
| | l | | | | | | | | | | | | | | | | | | | I | | O | |
| | m | | | | | | | | | | | | | | | | | | C | | | C | O |

Figure 7. The ISM matrix of the process of Fig. 3.

To facilitate comparison with the DSM, the ISM of Fig. 7 has been reduced to the matrix in Fig. 8 with input and output interfaces only.

| | | Interface (I, O) | | | | | | | | | | | | | | | | | | | |
|----------|---|------------------|---|---|---|---|---|---|---|----|----|----|----|----|----|---|--|--|--|--|--|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 8 | 9 | 11 | 13 | 15 | 17 | 18 | 20 | | | | | | |
| Activity | a | I | O | | | | | | | | | | | | | | | | | | |
| | b | | I | O | O | | | | | | | | | | | | | | | | |
| | c | | | I | O | | | | | | | | | | | | | | | | |
| | d | | | | I | O | | I | | | | | | | | | | | | | |
| | e | | | | | I | O | O | | | | | | | | | | | | | |
| | f | | | | | | I | O | O | | | | | | | | | | | | |
| | g | | | | | | | | C | | O | | | | | | | | | | |
| | h | | | | | | | | I | O | | | | | I | | | | | | |
| | l | | | | | | | | | I | O | | | | | | | | | | |
| | j | | | | | | | | | | C | | | | O | O | | | | | |
| | k | | | | | | | | | | | I | O | O | | | | | | | |
| | l | | | | | | | | | | | | | | I | | | | | | |
| | m | | | | | | | | | | | | | | | O | | | | | |

Figure 8. The reduced ISM matrix of Fig. 7.

Process decomposition is of importance in product life-cycle management, as it allows identification of a process structure, including the block-diagonal structure shown in Fig. 9. This matrix in Fig. 9 was obtained from the matrix in Fig. 8 by the cluster-identification algorithm (Kusiak and Chow 1987).

| | 0 | 1 | 2 | 3 | 4 | 5 | 8 | 9 | 17 | 18 | 11 | 13 | 15 | 20 |
|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| a | I | O | | | | | | | | | | | | |
| b | | I | O | O | | | | | | | | | | |
| c | | | I | | O | | | | | | | | | |
| d | | | | I | | O | I | | | | | | | |
| e | | | | | I | | | | | | | | | |
| f | | | | | | I | O | O | | | | | | |
| h | | | | | | | | I | | I | | | | |
| j | | | | | | | | | | O | O | | | |
| l | | | | | | | | | I | | | | | |
| g | | | | | | | | | | | O | | | |
| i | | | | | | | | | | | I | O | I | |
| k | | | | | | | | | | | | I | O | |
| m | | | | | | | | | | | | | | O |

Figure 9. The decomposed ISM of Fig. 8.

Each of the three blocks in Fig. 9 represents a group of activities that can be performed concurrently when necessary and the resources needed are available. The three blocks clearly represent the three sub-processes shown in Fig. 6 (S3, S2, and S1). In addition to the process block-diagonal structure, cycles can be identified by scanning the sub-matrices for the I-O patterns highlighted in Fig. 9. Such patterns can be determined with the modified cluster-identification algorithm (Kusiak and Chow 1987). Having identified the blocks and the cycles in each block, a topological-sorting algorithm organizes the ISM in Fig. 9 to produce the result equivalent to the DSM of Fig. 5 and the corresponding process model of Fig. 6.

The meta-algorithm, called here Meta-ISM, for organizing the ISM is as follows:

Step 1: Decompose the ISM with a clustering algorithm. Two cases of the ISM are possible:

Case 1: The block-diagonal structure is embedded in the ISM. The cluster-identification algorithm (Kusiak and Chow 1997) can be applied to decompose the ISM into mutually exclusive sub-matrices.

Case 2: The block-diagonal structure is not embedded in the ISM. The branch-and-bound algorithm (Kusiak 1999) is a natural choice for ISM decomposition.

Step 2: Identify cycles in each of the sub-matrices. This can be accomplished with the modified cluster-identification algorithm (Kusiak and Chow 1997). For two entries $(k, p) = I$ and $(k, q) = O$ in row k and columns p and q , the algorithm would need to identify row l with the corresponding entries $(l, p) = O$ and $(l, q) = I$. For example, the entries $(i, 13) = O$, $(i, 15) = I$, and $(k, 13) = O$, $(k, 15) = O$ in Fig. 9 represent a cycle.

Step 3: Replace cycles with aggregated activities. Though this step is not necessary, the activities involved in any cycle can be merged in an aggregate activity with the cycle-merging scheme presented in Kusiak (1999). One of the reasons for merging cycles is to simplify Step 4.

Step 4. Apply the topological-sorting algorithm to the matrix of the corresponding graph produced in Step 3.

The Meta-ISM presented produces a result that is more than equivalent to the result generated by the triangularization algorithm (Kusiak et al. 1994). The advantages of the proposed Meta-ISM algorithm are as follows:

- Efficiency and scalability. Though not formally proven in this paper, the four steps of the Meta-ISM algorithm are computationally efficient and scalable.
- Transparent and structured approach. The Meta-ISM algorithm involves a small number of steps (4), yet provides plenty of insights into the computation process.
- Ease of integration. This could be the single most important feature of the proposed Meta-ISM algorithm because it provides easy integration with the widely used data platforms. Many similar approaches published in the literature have not considered this aspect.

3 SUMMARY

The basic idea discussed in the paper focuses on understanding and reducing the complexity of systems, including activities involved in the product life-cycle management. There is likely one area of science aimed at efficiency improvement that applies to products, systems, and services. The paper sets the groundwork for uniformity of modelling and solution approaches across different applications. The DSM and ISM concepts were used to illustrate that modelling and solving the same problem can be considered from more than one perspective. The advantages and disadvantages of the two approaches examined in the paper were discussed. The paper was intended to consider different perspectives to modelling and solving problems that call for a systems-engineering approach.

4 REFERENCES

- [1] Bilalis, N., Maravelakis, E., Antoniadis, A., and Moustakis, V., 2004, Mapping product innovation profile to product development activities - The I-DSM tool, IEEE International Engineering Management Conference, Singapore, pp. 1018–1022.
- [2] 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, pp. 292–306.
- [3] Chen, C.H., Ling, F.S., and Chen, W., 2003, Project scheduling for collaborative product development using DSM, International Journal of Project Management, Vol. 21, No. 4, pp. 291–299.
- [4] Chen, K.M and Liu, R.J., 2005, Interface strategies in modular product innovation, Technovation, Vol. 25, No. 7, pp. 771–782.
- [5] Cho, S.H. and Eppinger, S.D., 2005, A simulation-based process model for managing complex design projects,

- IEEE Transactions on Engineering Management, Vol. 52, No. 3, pp. 316–328.
- [6] Jose, A. and Tollenaere, M., 2005, Modular and platform methods for product family design: Literature analysis, *Journal of Intelligent Manufacturing*, Vol. 16, No. 3, pp. 371–390.
- [7] Karniel, A., Belsky, Y., and Reich, Y., 2005, Decomposing the problem of constrained surface fitting in reverse engineering, *Computer-Aided Design*, Vol. 37, No. 4, pp. 399–417.
- [8] Kusiak, A., 1999, *Engineering Design: Products, Processes, and Systems*, Academic Press, San Diego, CA.
- [9] Kusiak, A. and Chow, W.S., 1987, An efficient cluster identification algorithm, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 17, No. 4, pp. 696–699.
- [10] Kusiak, A., Larson, N., and Wang, J., 1994, Reengineering of design and manufacturing processes, *Computers and Industrial Engineering*, Vol. 26, No. 3, pp. 521–536.
- [11] Lindemann, U., Danilovic, M., Deubzer, F., Maurer, M., and Kreimeyer, M., 2007, Proceedings of the 9th International DSM Conference, Munich, Germany.
- [12] Sosa, M.E., Eppinger, S.D., and Rowles, C.M., 2000, Design modular and integrative systems, Proceedings of the ASME 2000 International Design Engineering Technical Conference, Baltimore, MD, pp. 1–10.
- [13] Steward, D.V., 1981, The design structure system: A method for managing the design of complex systems, *IEEE Transactions on Engineering Management*, Vol. 28, pp. 71–74.
- [14] Steward, D.V., 1981, *Systems Analysis and Management: Structure, Strategy and Design*, Princeton, NJ: Petrocelli Books.
- [15] Şule, T. P. and Mustafa, P., 2006, Modelling detailed information flows in building design with the parameter-based design structure matrix, *Design Studies*, Vol. 27, No. 1, pp. 99–122.