23rd INTERNATIONAL DEPENDENCY AND STRUCTURE MODELING CONFERENCE, DSM 2021
MONTRÉAL, CANADA, 12 – 14 October, 2021

Interface-Component Model (I-CM) as a system design tool for an SE toolset with DSM

Yordan Tuzsuzov[1]

[1]Tuzsuzov, Germany

Abstract: DSMs are a simple and powerful tool to represent and work with system models. However, the representation of the interfaces as anonymous pairing entities significantly limits its usage as a stand-alone system design tool. As a result, DSMs are typically engaged in the engineering stage, after the system has been already designed – mostly to cluster, sequence, or analyze the complexity of the system. This paper presents the Interface-Component Model (I-CM) as a method that models systems in a similar simple matrix fashion as a DSM, N2 Matrix, QFD, etc., but resolves some significant limitations of DSMs in the context of system design. With the algorithm to export the system model from an I-CM to a DSM, both tools may be used together as an ecosystem for system design and analysis based on matrix methods.

Keywords: Interface-Component Model, I-CM, Systems Engineering, System Design, DSM, DMM, System Complexity, Instant Dependency Highlight

1 Interface Component Model

Considering the system as a group of *interconnected elements* with a certain *function* or *purpose* (Meadows, 2008), the I-CM (Tuzsuzov, 2020) sets the elements (components) and the interconnections (interfaces) as two dimensions of the whole. The I-CM treats the components and the interfaces equally – each group owns one dimension of the system model (Figure 1). On the intersections between the components and their interfaces within the matrix, special notation is used to identify the type of flow as "I", "O", or "IO" (input, output, and I/O, respectively). This may be applied to information, energy, and/or material flows. In addition, other notations may be used, like "S" for static, "M" for mechanical, etc. These are considered always bi-directional.

Components → / Interfaces ↓	Part 1	Part 2	Part 3	Part 4	Part 5
Interaction 1	I		O		
Interaction 2		IO	IO	I	O
Interaction 3		O			I
Interaction 4	IO		IO		

Figure 1. I-CM of a system with five components and four interfaces

From the example in Figure 1, we can already identify some aspects that the standard binary DSM cannot address (E. Crawley et al., 2016; Tuzsuzov, 2020):

- Identity and purpose of each interface (the name of the interface indicates its purpose)
- Multiple interfaces with multiple purposes between components (e.g., *Part 1* and *Part 3*)
- One shared interface between multiple components (e.g., *Interaction 2*)

As a result of the limitations shown above, the DSM may give a wrong impression of the system structure and complexity (Sinha et al., 2018). Let's look at the example of six computers communicating via network bus (Figure 2). We can model this system using the I-CM as shown in Figure 3. This system has low complexity, and adding more nodes to the network would not principally increase its technical complexity.

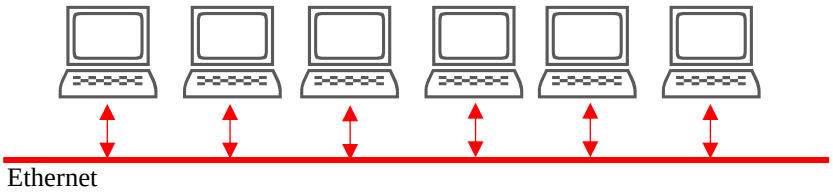

Figure 2. A computer network bus with six nodes

Display open dependencies Components → ↓ Interfaces	Computer 1	Computer 2	Computer 3	Computer 4	Computer 5	Computer 6		Interface connectivity
Ethernet	IO	IO	IO	IO	IO	IO		6
								0
Component connectivity:	1	1	1	1	1	1	0	

Figure 3. I-CM of the network system

However, the DSM of this setup shows a highly interconnected and complex system, in contrast to the reality (Figure 4). In fact, the DSM of this system would not differ from the DSM of a system of six computers with P2P lines from each computer to each of the others.

	Computer 1	Computer 2	Computer 3	Computer 4	Computer 5	Computer 6
Computer 1		X	X	X	X	X
Computer 2	X		X	X	X	X
Computer 3	X	X		X	X	X
Computer 4	X	X	X		X	X
Computer 5	X	X	X	X		X
Computer 6	X	X	X	X	X	

Figure 4. DSM of the computer network system, generated from the I-CM

A possible way to optimize the presentation in the DSM is to consider the Ethernet carrier as a component itself, and not as an interface. The DSM following this approach is shown in Figure 5. Although this representation is closer to the reality as far as the complexity is concerned, it lacks the logical separation of interfaces and components that is so much needed in the system design.

	Computer 1	Computer 2	Computer 3	Computer 4	Computer 5	Computer 6	Ethernet
Computer 1							X
Computer 2							X
Computer 3							X
Computer 4							X
Computer 5							X
Computer 6							X
Ethernet	X	X	X	X	X	X	

Figure 5. Considering the Ethernet as “Component” instead of “Interface”

There is also the opposite example, where the DSM system representation indicates less complexity than the reality. This could be demonstrated with two components/subsystems having multiple interfaces for different purposes. Part of a spacecraft (space laboratory) architecture is shown graphically in Figure 6. The respective I-CM (Figure 7) shows the same level of detail and gives a good indication of the system complexity. The DSM, however, (Figure 8) will reveal a quite simple system, except we don't code each of the interfaces as components (as we have done in the previous example – Figure 5).

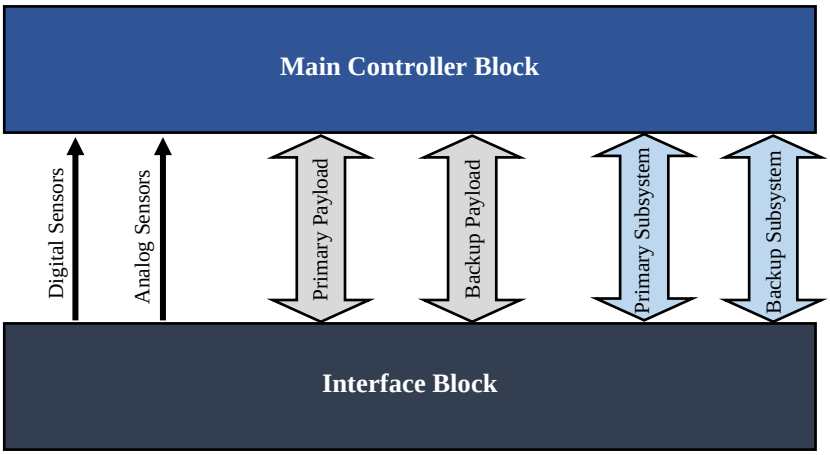

Figure 6. Part of spacecraft E/E architecture

Display open dependencies Components → Interfaces ↓	Main Controller Block	Interface Block		:Interface connectivity
Primary Payload Bus	IO	IO		2
Backup Payload Bus	IO	IO		2
Primary Subsystem Bus	IO	IO		2
Backup Subsystem Bus	IO	IO		2
Digital Sensors	I	O		2
Analog Sensors	I	O		2
				0
Component connectivity:	6	6	0	

Figure 7. I-CM of the part of the E/E architecture of a spacecraft

	Main Controller Block	Interface Block
Main Controller Block		X
Interface Block	X	

Figure 8. DSM of the part of the E/E architecture of a spacecraft

So far, we have demonstrated the approach of the I-CM to describe the physical system consisting of *elements* and their *interconnections* (components and interfaces). To cover all three aspects of the system, the I-CM tool uses a second matrix to map the *purpose*/*functions* to the physical system. This second matrix is in fact a variant of the already-known Form-to-Function Mapping or Domain Mapping Matrix (DMM) (Danilovic and Browning, 2007). As both the I-CM and DMM matrices have the components in the columns, they can be aligned on that dimension.

To demonstrate how to model the system with all *elements*, *interconnections*, and *purposes* in one document, we take an example of a radio receiver. The purpose of the radio receiver is to play radio transmissions in AM and FM (MW and VHF bands). The selection of the band, stations, and sound volume should be done with a simple graphical UI and controlled with an integrated keyboard. The graphical diagram of its electrical architecture is shown in Figure 9.

Figure 10 shows the I-CM of the system and its assigned purpose derived from the above description. The numbers in the DMM matrix indicate the logical sequence of the function deployment over the physical system (e.g., the "reception and play" function is following the path antenna→tuner→amplifier→speaker), but an "X" could be used instead when sequencing is not needed or wanted (for parallel processes, feedback loops, etc.).

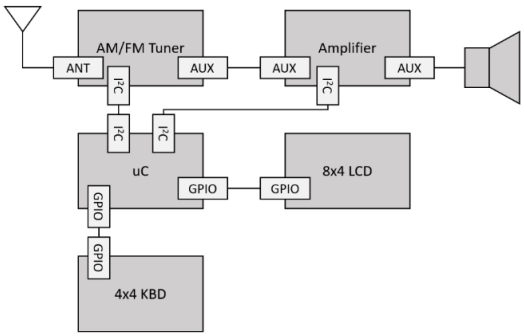

Figure 9. Electrical architecture of the AM/FM radio receiver

Display open dependencies (Components → / Interfaces ↓)	Antena	AM/FM Tuner	Digital Amplifier	microController	LCD	4x4 Keyboard	Speaker	
AM/FM Electrical Signal	**O**	**I**						
Analog Audio Signal (tuner)		**O**	**I**					
I2C command line tuner		**IO**		**IO**				
I2C command line amplifier			**IO**	**IO**				
8xGPIO for LCD control				**IO**	**IO**			
8XGPIO for KBD input				**IO**		**IO**		
Analog Audio Signal (amplified)			**O**				**I**	

FUNCTION / REQUIREMENT	Component Interactions							
Receive and play radio broadcast programs in AM(535-1605 kHz) and FM(77-108MHz) bands	1	2	3				4	
Allow the user to select the band and the station via keyboard and display it on an LCD;		3		2	3	1		
Allow the user to select the audio volume via keyboard and display it on an LCD;			3	2	3	1		

Figure 10. I-CM of the AM/FM radio receiver

2 Instant relation highlight: The main feature of I-CM

2.1 In addition to the integrated matrix system representation, the I-CM tool has one multipurpose function – instantly highlighting the system dependencies. This can be used for impact analysis, sanity checks, design- and integration-planning, minimal viable system definition, and more. Figure 11 shows the highlighted subsystem that delivers the “band and station selection” function of our radio.

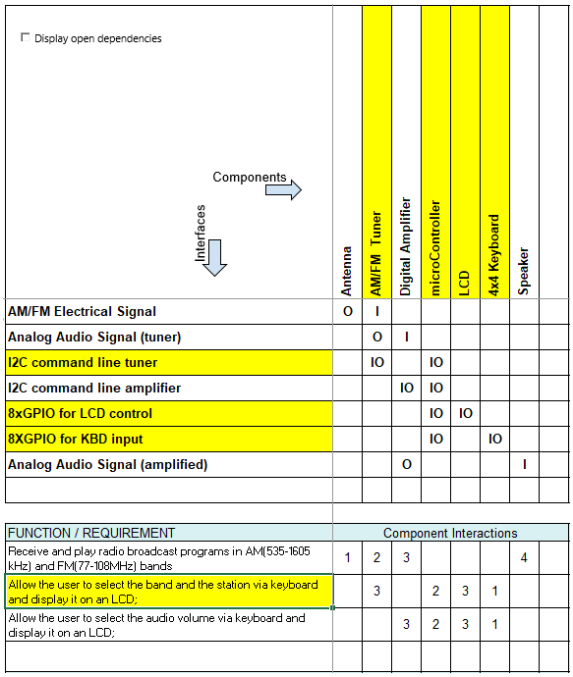

Figure 11. Highlighted subsystem for “band and station selection” function

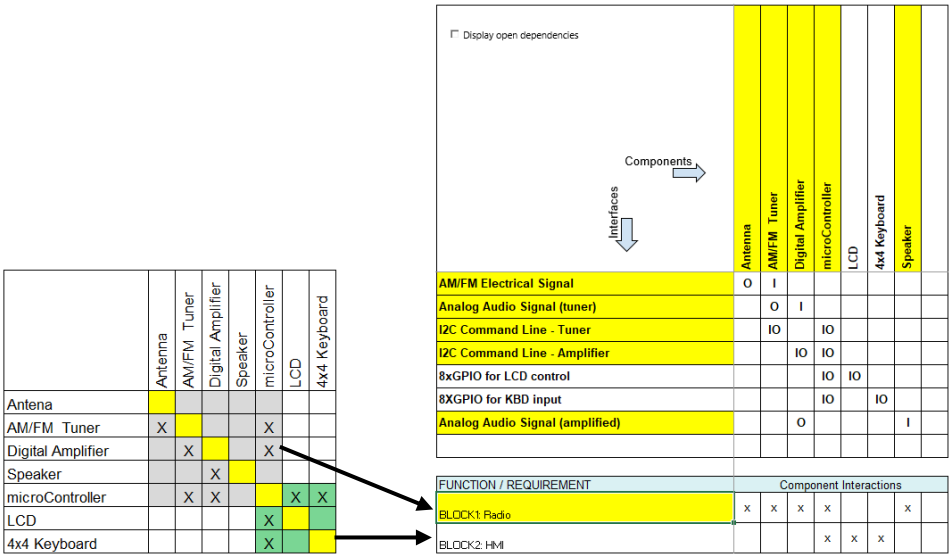

Figure 12. Domain mapping of clustering from DSM.

3 Exchange between I-CM and DSM

The I-CM contains more information than a binary DSM, which is why the I-CM tool supports export of its model to a DSM. The DSM can then be used to cluster and sequence the system.

The automatic import from DSM, however, could lead to loss of data. The matrix would need to be manually sequenced and clustered after import, but this could then be used as "clustering by purpose," as shown in Figure 12. The clusters identified by the DSM are converted to a domain in the DMM part of the I-CM. The highlighted cluster "BLOCK1: Radio" corresponds to the first cluster of the DSM. As seen in this example, not only the interfaces, but also the clusters have the name and purpose visualized.

4 Conclusion

I-CM provides a simple and effective way to represent, analyze, and manipulate systems. It addresses some shortcomings of DSMs as a system design tool while keeping the familiar matrix notation. However, as each method can better address different aspects of the engineering processes, they can easily complement each other to form a powerful toolset for systems engineering.

References

Danilovic, M., Browning, T.R., 2007. Managing complex product development projects with design structure matrices and domain mapping matrices. International Journal of Project Management 25, 300–314. https://doi.org/10.1016/j.ijproman.2006.11.003.

E. Crawley, B. Cameron, Daniel Selva, 2016. System Architecture: Strategy and Product Development for Complex Systems. Pearson, Boston.

Meadows, D.H., 2008. Thinking in systems: A primer. Chelsea Green Publishing, London, 218 pp.

Sinha, K., Suh, E.S., Weck, O. de, 2018. Integrative Complexity: An Alternative Measure for System Modularity. Journal of Mechanical Design 140, 051101. https://doi.org/10.1115/1.4039119.

Tuzsuzov, Y., 2020. Systems Engineering for All: Introduction to Systems Engineering for non-Systems Engineers. tredition.

Contact: Y. Tuzsuzov, yt3007@rit.edu