ESD.33 Systems Engineering

Lecture 6

Requirements Driven Systems Design

Qi Van Eikema Hommes

Course Layout

Lecture 4
Stakeholder Analysis and Requirements Definition

Lecture 2
Systems Engineering As Human Activity

Lecture 13
Design Verification and Validation, Lifecycle Management

Lecture 5 innovation in Systems Engineering

Lecture 6 Axiomatic Design and DM-DSM Method

Lecture 7 CPM and DFSS

Lecture 8
Trade Space Exploration Concept Selection

Lecture 10: Experiments
Lecture 11: Robust Design I
Lecture 12: Robust Design II

Lecture Outline

- Introduction to Axiomatic Design
  - Four domains
  - Axiom 1—Independence Axiom
    - Design Matrix
    - Zigzagging
    - Constraints
  - Axiom 2—Information Axiom
- Design Structure Matrix for Technical Systems
- DM—DSM Method

The Founder of Axiomatic Design Theory

- **Nam Pyo Suh**—MIT Professor Emeritus.
- B.S., Mechanical Engineering, 1959, M.S., Mechanical Engineering, 1961, MIT
- Ph.D, Mechanical Engineering, 1964, Carnegie Mellon University.
- From 1965-1969, Suh served as a professor at the University of South Carolina. In 1970 he began his professional career at MIT-- serving as director of the MIT-Industry Polymer Processing Program from 1973-1984; director of the Laboratory for Manufacturing and Productivity from 1977-1984; and **Mechanical Engineering Department Head from 1991 to 2001**. Although still keeping the title of Ralph E. Cross Professor of Mechanical Engineering at MIT, Suh is now president of KAIST.

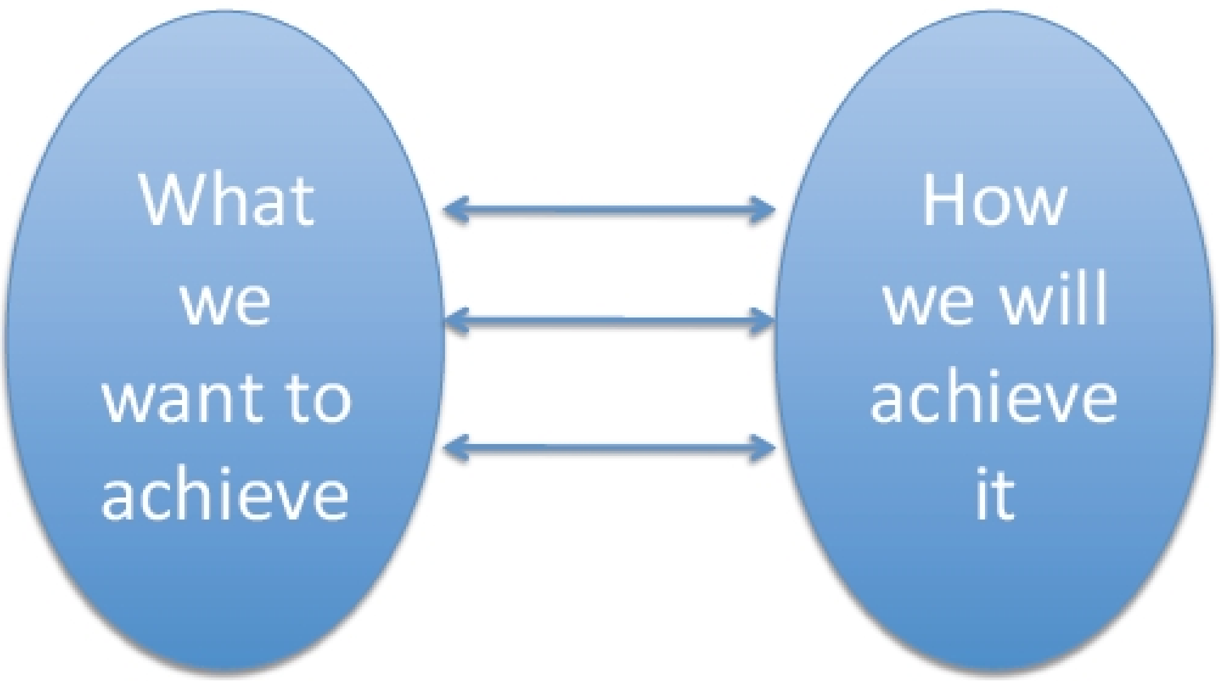
The Goals of Axiomatic Design

- Establish a **scientific basis** for design
- Improve design activities by providing the designer with a theoretical foundation based on **logical and rational thought processes** and tools.
- Make human designers **more creative**
- **Reduce** the random search process
- **Minimize** the iterative **trial and error** process
- Determine **the best designs** among those proposed

Suh, Axiomatic Design, 2000, page 5

Definition of Design

- Design is an interplay between what we want to achieve and how we will achieve it.



The Four Domains of Design

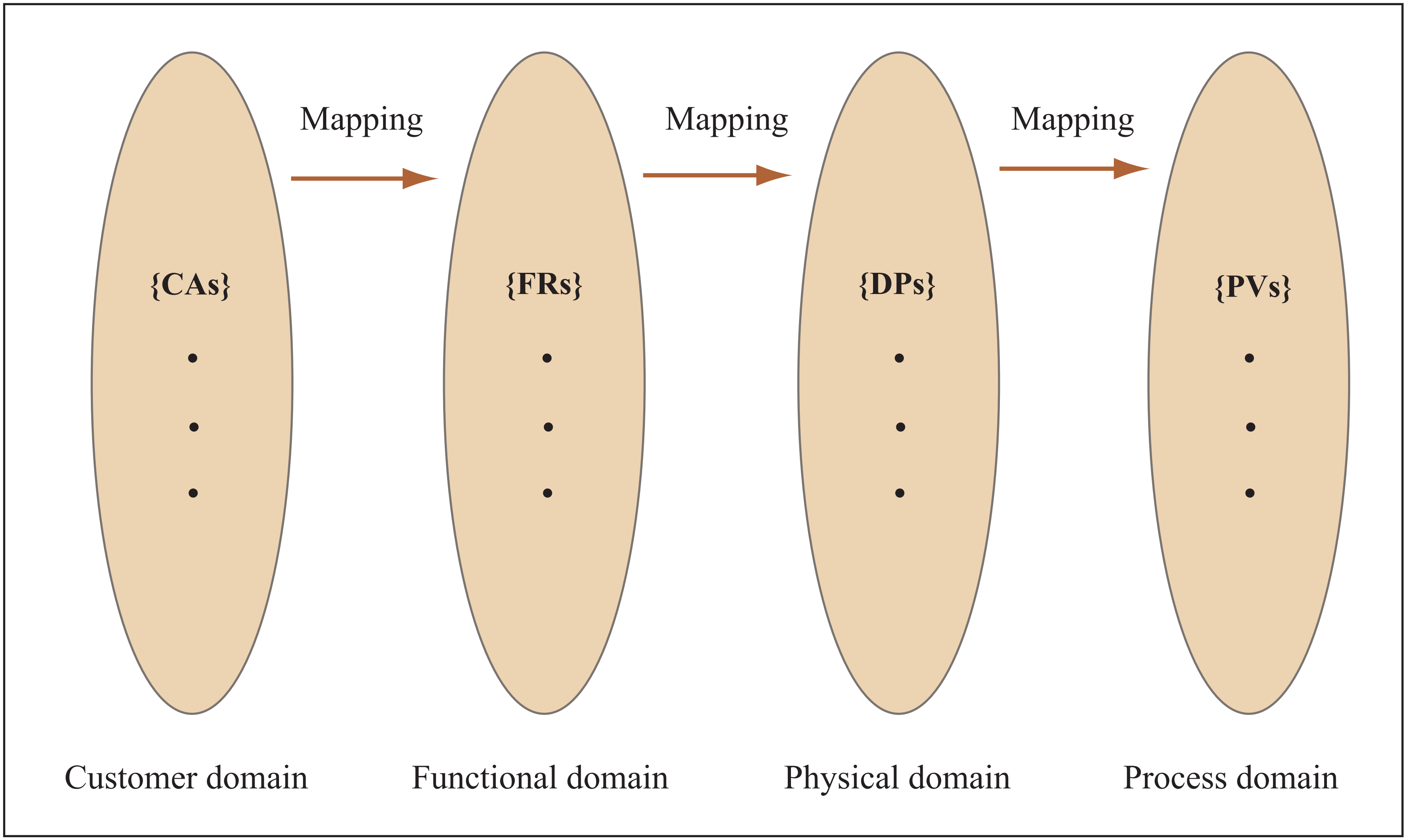


Image by MIT OpenCourseWare.

Definitions

- **Customer Attribute (CA)**—what customer desire from a product
- **Functional Requirement (FR)**—minimum set of independent requirements that completely characterize the functional needs of the product in the functional domain.
- **Design Parameter (DP)**—Key physical variables in the physical domain that characterize the design that satisfies the specified FRs.
- **Process Variables (PV)**—key variables in the process domain that characterize the process that can generate the specified DPs.

Benefits of the Domains

- Customer Needs are stated in the customer's language
- Functional Requirements and Constraints are determined to satisfy Customer Needs
- "The FRs must be determined in a solution neutral environment" (or, in other words, say "what" not "how")
  - BAD = the adhesive should not peel
  - BETTER = the attachment should hold under the following loading conditions
- Provide Requirements Traceability

Lecture Outline

- ❑ Introduction to Axiomatic Design
  - ✓ Four domains
  - ❑ Axiom 1—Independence Axiom
    - ❑ Design Matrix
    - ❑ Zigzagging
  - ❑ Axiom 2—Information Axiom
- ❑ Design Structure Matrix for Technical Systems
- ❑ DM—DSM Method

Axiom

- Axioms are truths that cannot be derived but for which there are no counter examples or exceptions.
- Examples of Axioms:
  - First and second law of thermodynamics
  - Newton’s three law of mechanics

How were the Design Axioms Created?

- Identifying the common elements that are present in all good designs:
  - How did I make such a big improvement in a process?
  - How did I create the process?
  - What are the common elements in good designs?
- Use logical reasoning process to reduce the observations to **two Axioms**.

The Two Axioms

- **Axiom 1: Independence Axiom**—maintain the independence of functional requirements (FRs).
- **Axiom 2: The Information Axiom**—minimize the information content of the design.

Design Matrix [A]

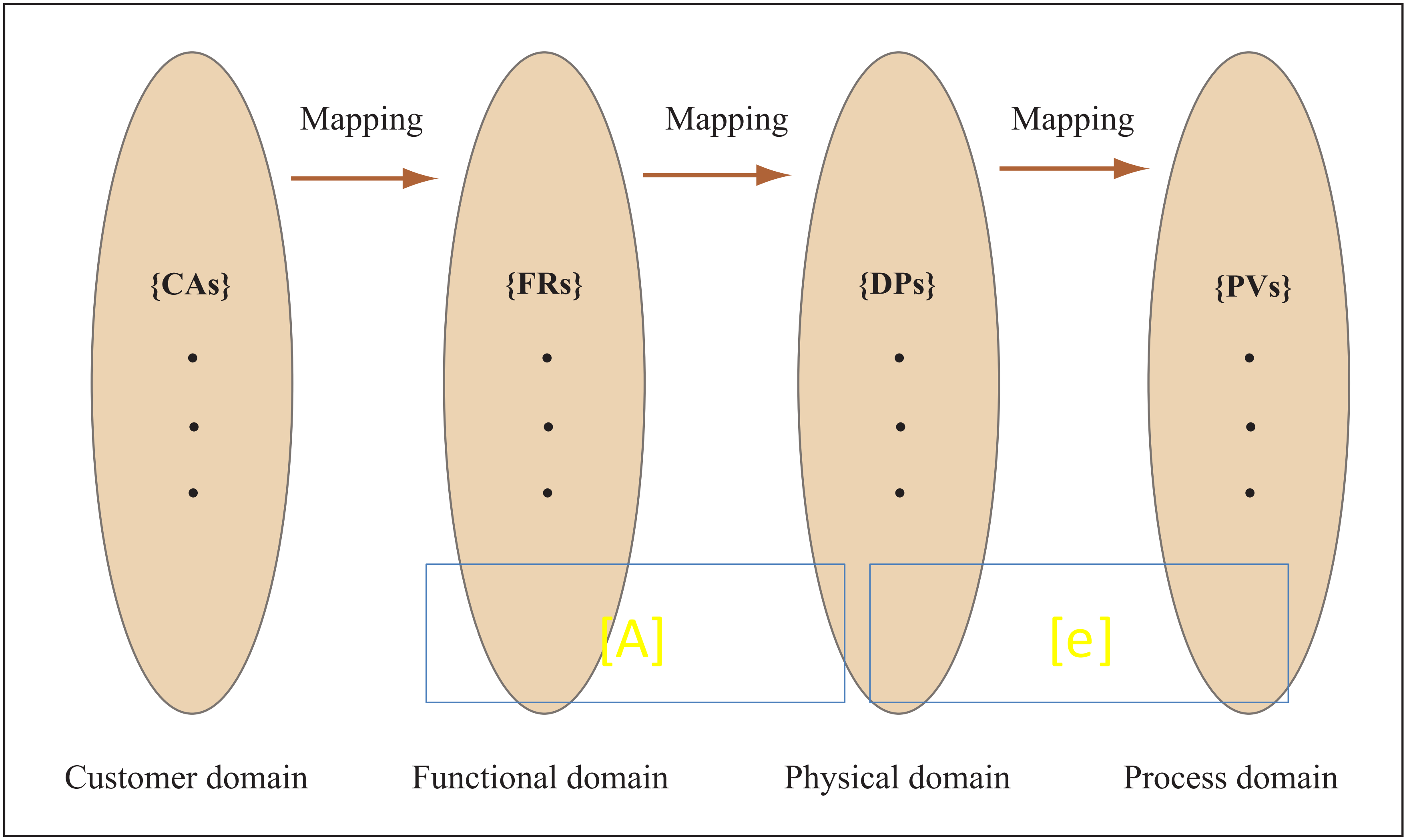


Image by MIT OpenCourseWare.

Design Matrix

{FR} = [A] {DP}

$$\begin{Bmatrix} FR_1 \\ FR_2 \\ FR_3 \end{Bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \bullet \begin{Bmatrix} DP_1 \\ DP_2 \\ DP_3 \end{Bmatrix}$$

Design Matrix Example

- FR1 = Provide access to the items stored in the refrigerator
- FR2 = Minimize energy loss
- DP1 = Vertically hung door
- DP2 = Thermal insulation material in the door

Image by MIT OpenCourseWare.

$$\begin{Bmatrix} FR1 \\ FR2 \end{Bmatrix} = \begin{Bmatrix} x & 0 \\ x & x \end{Bmatrix} \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix}$$

Suh, Axiomatic Design, 2000

A Different Design

- FR1 = Provide access to the items stored in the refrigerator
- FR2 = Minimize energy loss
- DP1 = Horizontal door
- DP2 = Thermal insulation material in the door

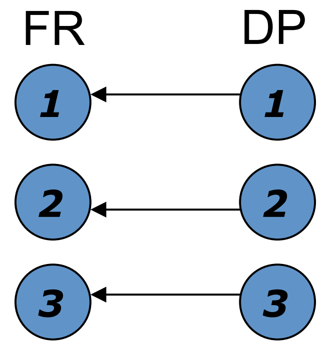
$$\begin{Bmatrix} FR1 \\ FR2 \end{Bmatrix} = \begin{bmatrix} x & 0 \\ 0 & x \end{bmatrix} \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix}$$

Image by MIT OpenCourseWare.

Three Types of Design Matrices

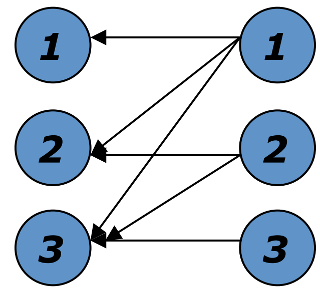
Uncoupled Design

$$\begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{bmatrix}$$



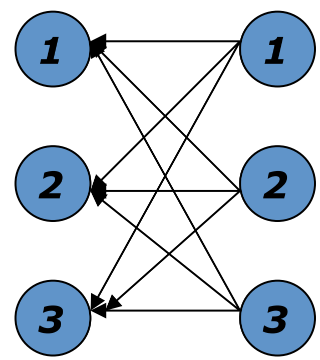
Decoupled Design

$$\begin{bmatrix} A_{11} & 0 & 0 \\ A_{21} & A_{22} & 0 \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$



Coupled Design

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$



Axiom 1: Independence Axiom

- To satisfy the Independence Axiom, the design matrix must be either diagonal or triangular.

$$\begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{bmatrix}$$

Uncoupled Design

$$\begin{bmatrix} A_{11} & 0 & 0 \\ A_{21} & A_{22} & 0 \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

Decoupled Design

Water Faucet Example

- Functional Requirements:
  - FR1: Adjust the water temperature (T)
  - FR2: Adjust the water volume (Q)

What is the Design Matrix?

Image by MIT OpenCourseWare.

What is the Design Matrix?

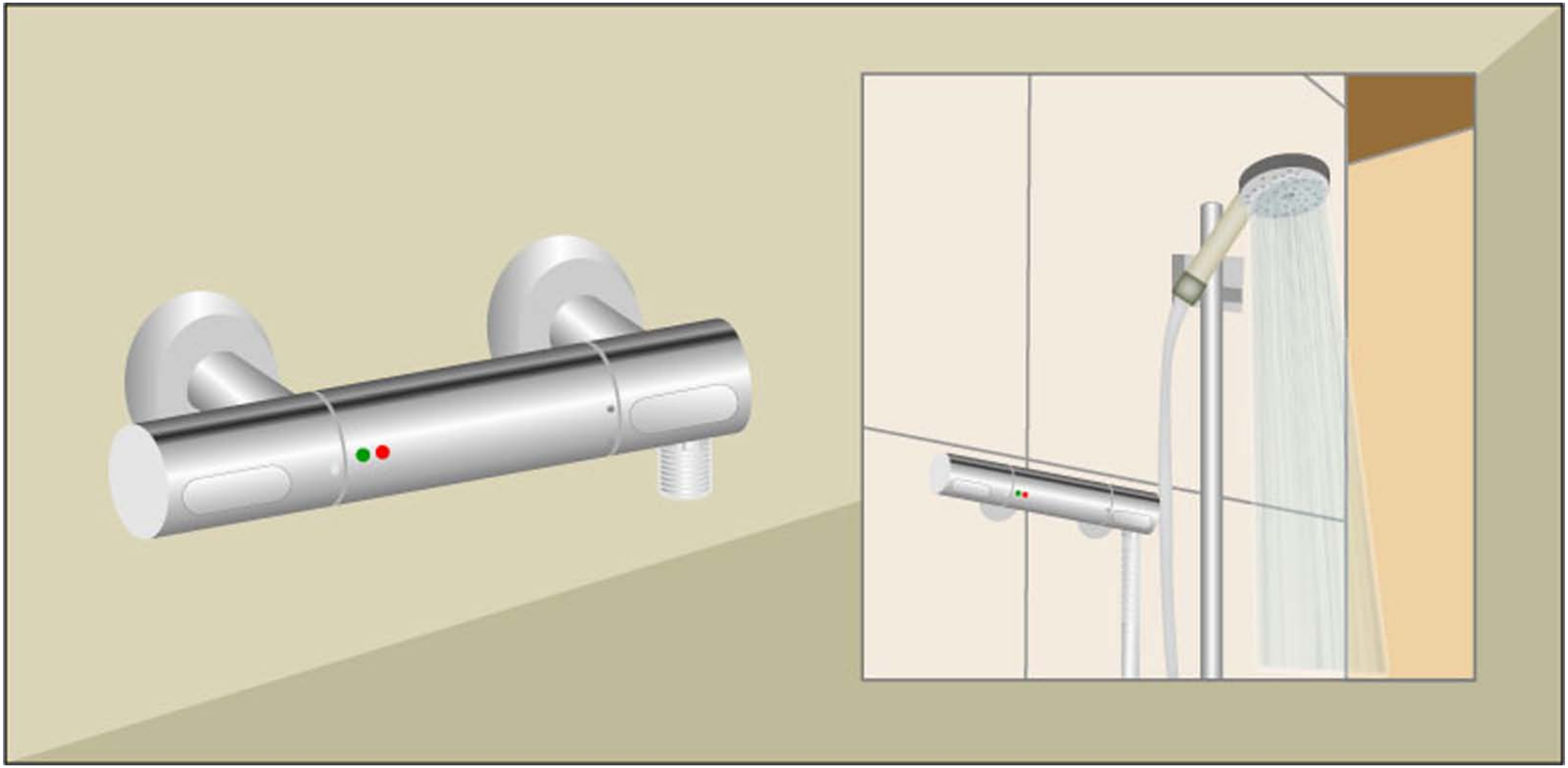


Image by MIT OpenCourseWare.

What is the Design Matrix?

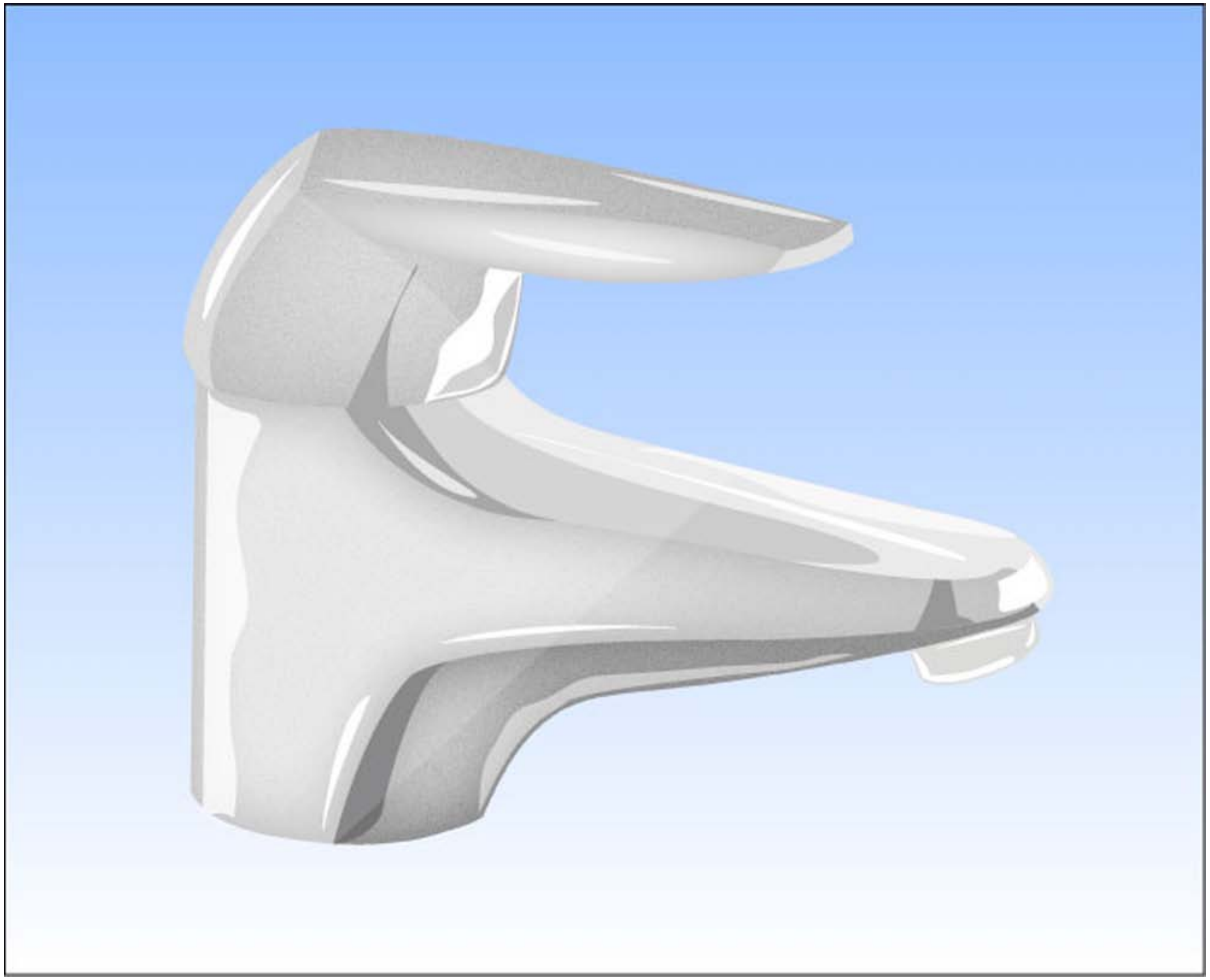
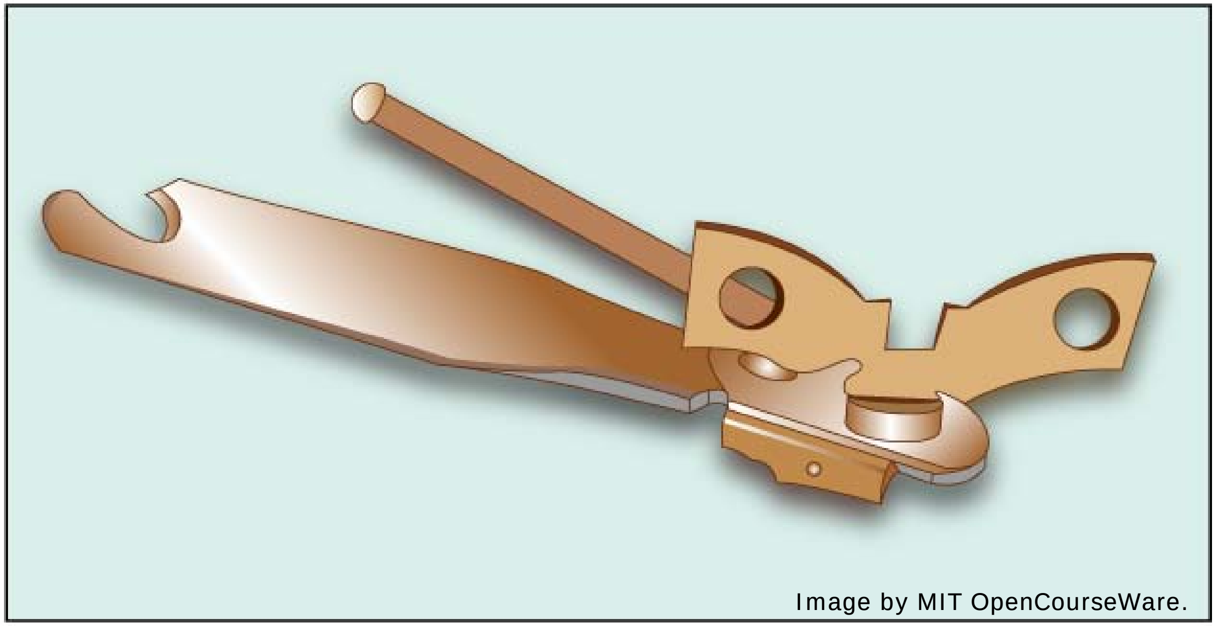


Image by MIT OpenCourseWare.

Functional Coupling vs Physical Coupling



# of parts ≠ # of DPs

Why Meeting Axiom 1 is Desirable?

Lecture Outline

- ❑ Introduction to Axiomatic Design
  - ✓ Four domains
  - ❑ Axiom 1—Independence Axiom
    - ✓ Design Matrix
    - ❑ Zigzagging
    - ❑ Constraints
  - ❑ Axiom 2—Information Axiom
- ❑ Design Structure Matrix for Technical Systems
- ❑ DM—DSM Method

Zig Zagging

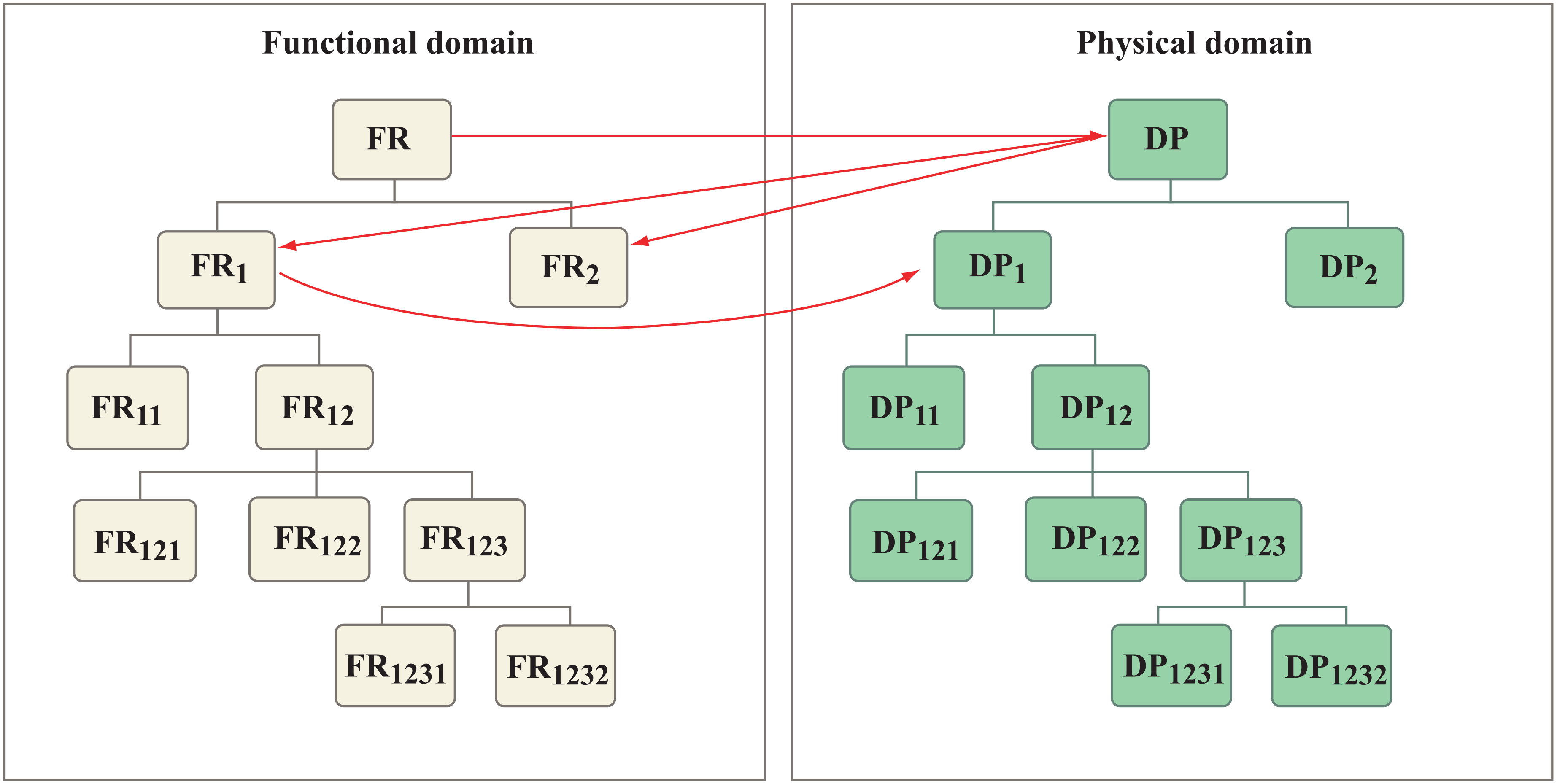


Image by MIT OpenCourseWare.

Refrigerator Design Example

- FR1 = Freeze food for long-term preservation
- FR2 = Maintain food at cold temp for short-term preservation

- DP1 = the freezer section
- DP2 = the chiller (refrigerator) section


Image by MIT OpenCourseWare.

$$\begin{Bmatrix} FR1 \\ FR2 \end{Bmatrix} = \begin{Bmatrix} x & 0 \\ 0 & x \end{Bmatrix} \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix}$$

Decompose the System

FR1 = Freeze food for long term preservation

- FR11 = Control freezer temp
- FR12 = Maintain uniform freezer temp
- FR13 = Control freezer humidity

FR2 = Maintain food at cold temp for short term preservation

- FR21 = Control chiller temp
- FR22 = Maintain uniform chiller temp

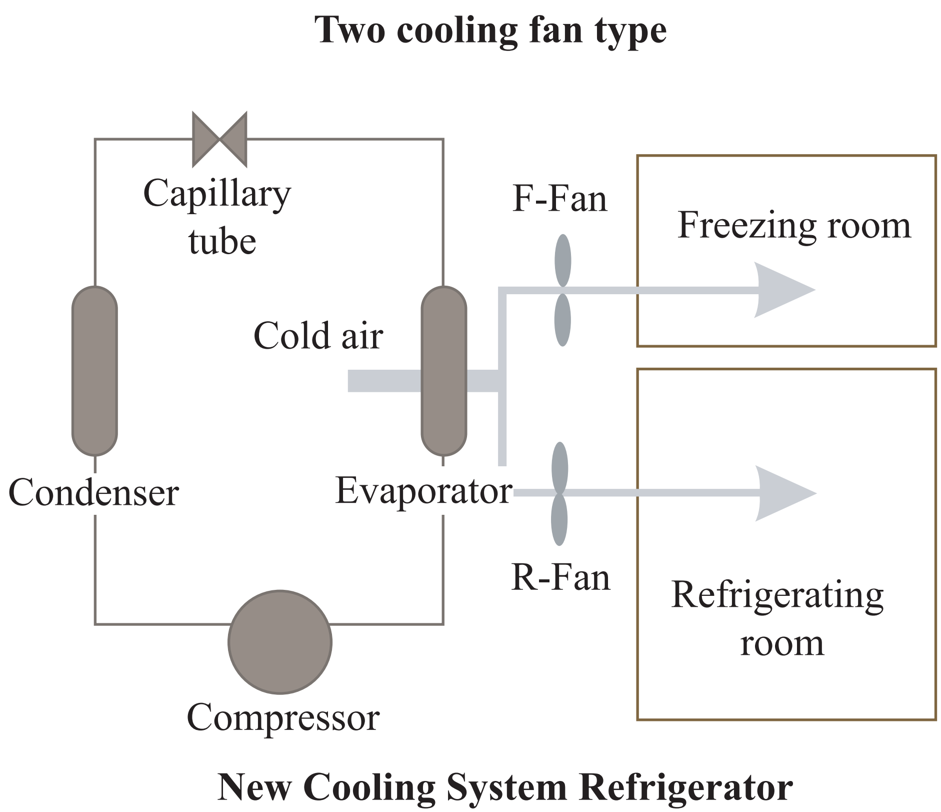
DP1 = The freezer section

- DP11 = Sensor/compressor system for freezer section
- DP12 = Air circulation system for freezer section
- DP13 = Condenser that condenses the moisture in the air when dew point is exceeded

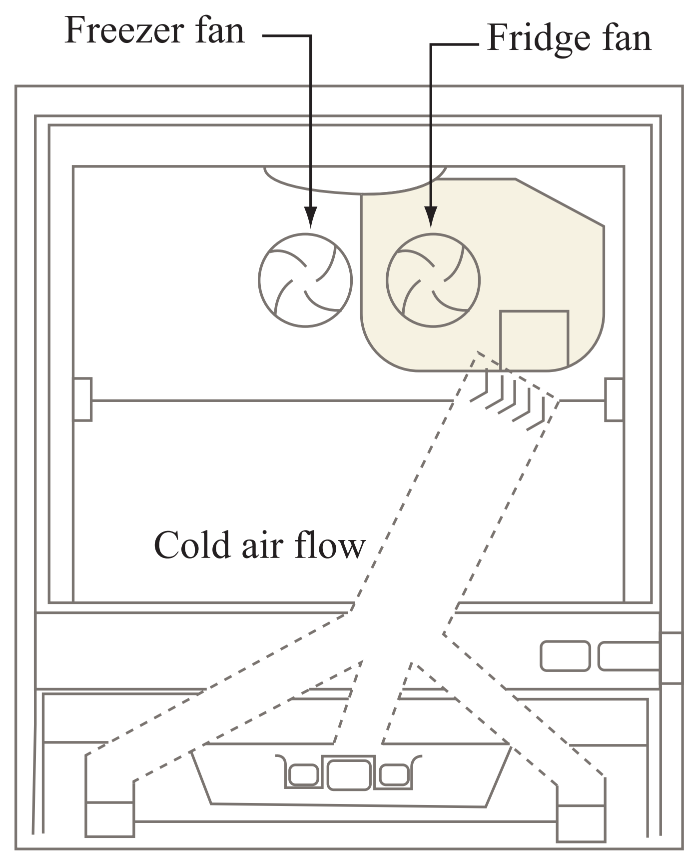
DP1 = The chiller section

- DP21 = Sensor/compressor for chiller section
- DP22 = Air circulation system for chiller section

What Does The Design Matrix Look Like?



A schematic drawing of a new refrigerator design.



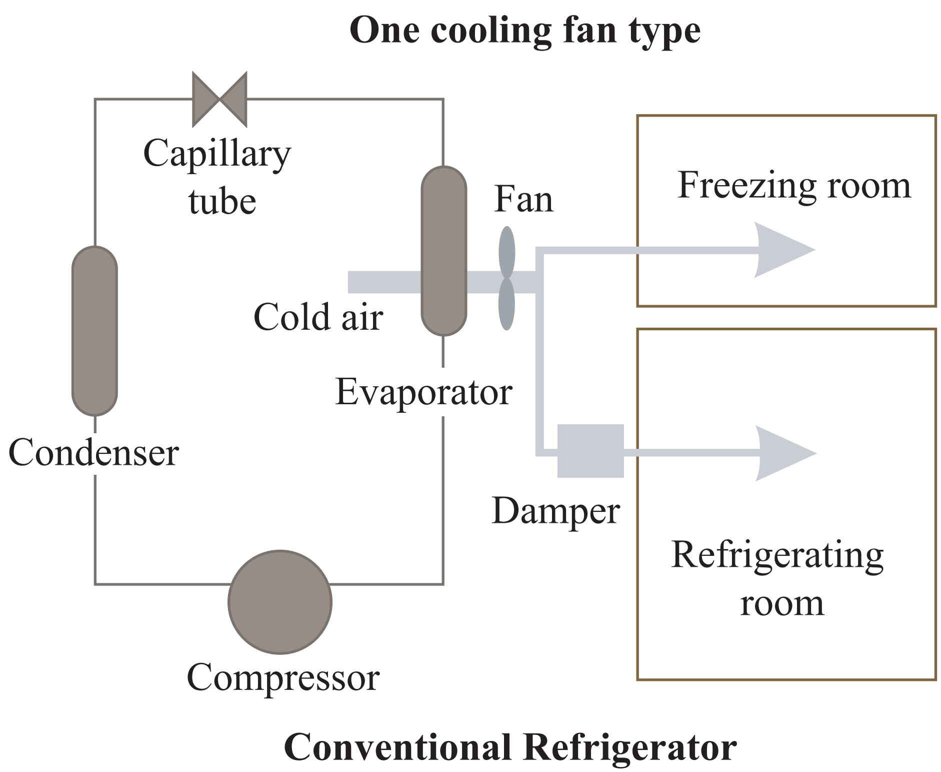
A cold-air circulation system with many vents.

Image by MIT OpenCourseWare.

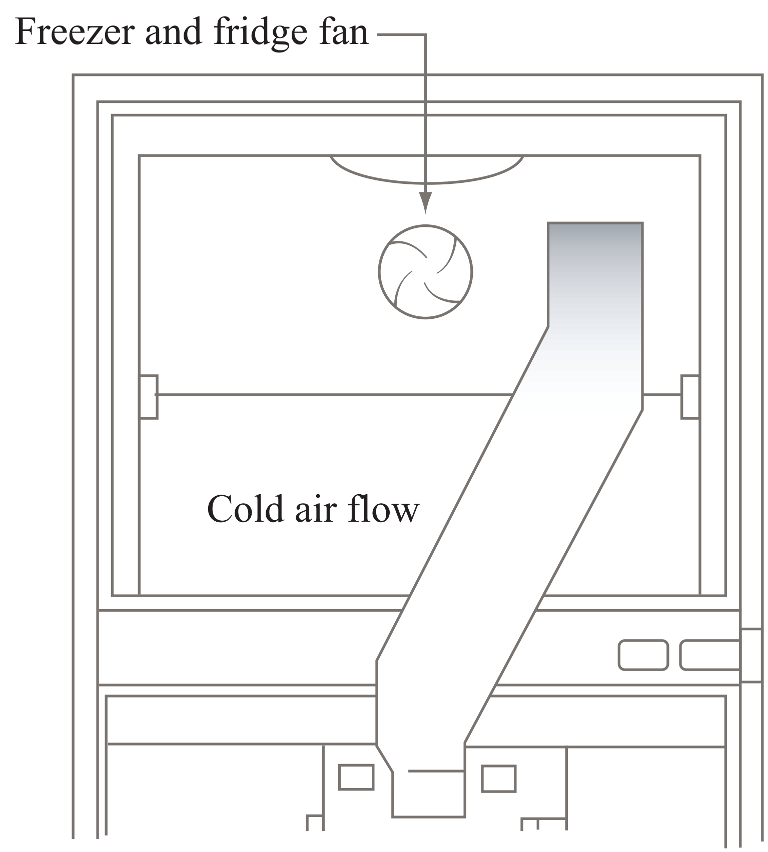
Design Matrix

		DP1			DP2	
		DP12	DP11	DP13	DP22	DP21
FR1	FR12	x	0	0	0	0
	FR11	x	x	0	0	0
	FR13	x	0	x	0	0
FR2	FR22	0	0	0	x	0
	FR21	0	0	0	x	x

Can We Save the Cost of a Fan?



Schematic drawing of a conventional refrigerator.



Cold-air circulation in a conventional refrigerator..

Image by MIT OpenCourseWare.

		DP1			DP2	
		DP12	DP11	DP13	DP22	DP21
FR1	FR12	x	0	0	0 X	0
	FR11	x	x	0	0 X	0
	FR13	x	0	x	0	0
FR2	FR22	0	0	0	x	0
	FR21	0 X	0 X	0	x	x

Coupled design!

Benefits So Far from Axiom 1

- Reduce system coupling early on.
- Start the design with requirements first.
- Think about the design concept first before applying robust engineering or optimization blindly.
- Zig-zagging instead of staying in one domain.
- Requirements traceability and rationale.

Class Discussions

- How does Zig-zagging help design synthesis?
- How does your organization decompose systems and requirements?
- Does this help with requirements traceability throughout the design?
- How does Axiomatic Design differ from QFD?

Lecture Outline

- ❑ Introduction to Axiomatic Design
  - ✓ Four domains
  - ❑ Axiom 1—Independence Axiom
    - ✓ Design Matrix
    - ✓ Zigzagging
    - ❑ Constraints
  - ❑ Axiom 2—Information Axiom
- ❑ Design Structure Matrix for Technical Systems
- ❑ DM—DSM Method

Constraints in Axiomatic Design

- **Constrant (C)**—are bounds on acceptable solutions. *Input constraints* are imposed as part of the design specification. *System constraints* are constraints imposed by the system in which the design solution must function.

Constraints

- Two types of constraints:
  - Input constraints—specific to the overall design goals (all design proposed must satisfy these).
    - Example: cost
  - System constraints—specific to a given design (they are the result of design decisions made).
    - Example: Diesel engine→ tailpipe emission standards for diesel engines
- What kind of constraint is Safety?

What Axiomatic Design Says about Constraints

- "Constraints provide bounds on the acceptable design solutions and differ from the FRs in that **they do not have to be independent**."

Lecture Outline

- ❑ Introduction to Axiomatic Design
  - ✓ Four domains
  - ✓ Axiom 1—Independence Axiom
    - ✓ Design Matrix
    - ✓ Zigzagging
  - ❑ Axiom 2—Information Axiom
- ❑ Design Structure Matrix for Technical Systems
- ❑ DM—DSM Method

Information Content

- Information Content I_i for a given FR_i is defined in terms of the probability P_i of satisfying FR_i:

  $I_i = log_2(1/P_i) = - log_2(P_i)$

- When there are m *FRs,*

$$I_{sys} = -\log_2(P_m) = -\sum_{i=1}^{m} \log_2 P_i$$

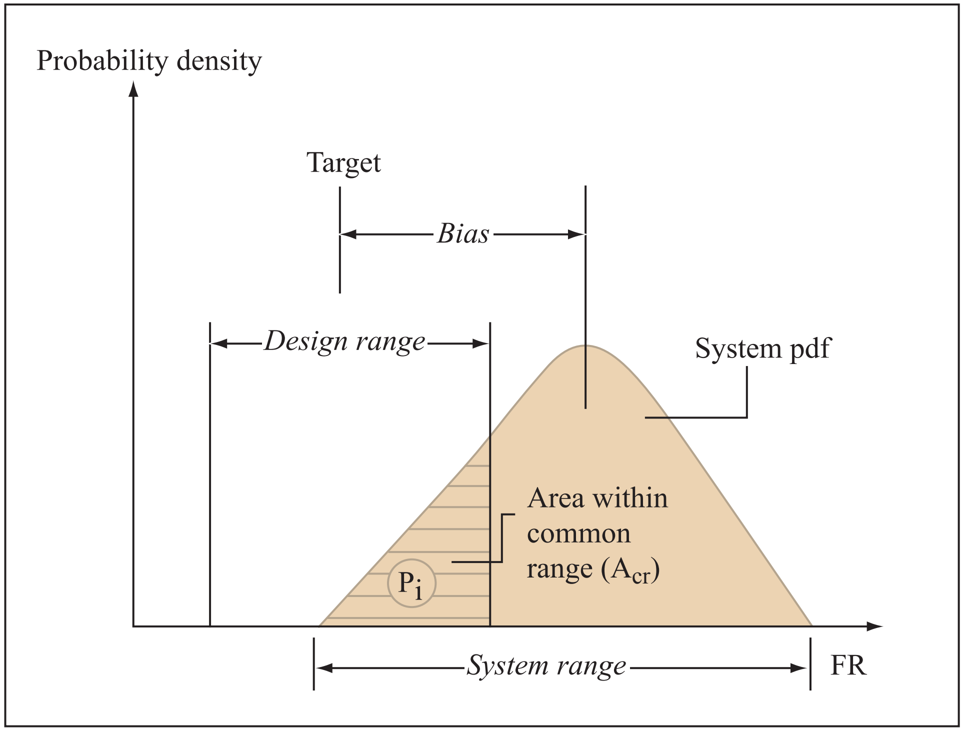


Image by MIT OpenCourseWare.

Axiom 2 Information Content

- The Information Axiom—Minimize information content *I*.
- Maximize the probability of meeting FRs.

$$I_{sys} = -\log_2(P_m) = -\sum_{i=1}^{m} \log_2 P_i$$

Example of Buying a House

Suh, Axiomatic Design, 2001

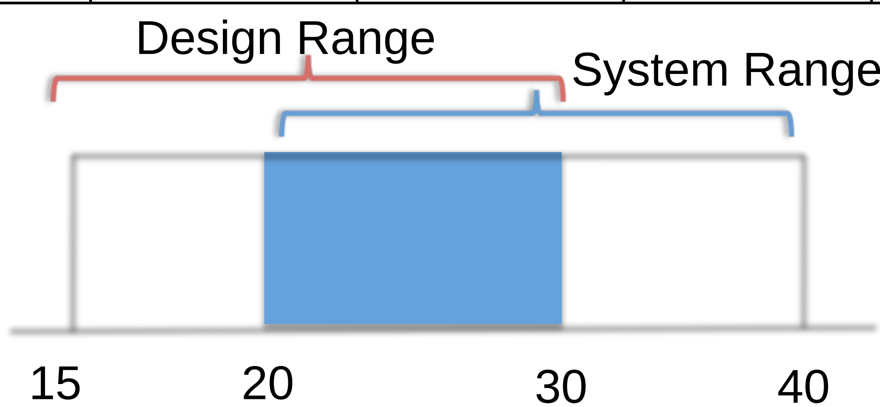
- FR1: Commute time 15 – 30 minutes
- FR2: Quality of School (65% or more highschool graduates go to colleges)
- FR3: Quality of air is good over 340 days a year
- FR4: price of house (4 BR, 3000 ft^2, less than 650K)

Town	FR1 = commute time (min)	FR2=Quality of schools (%)	FR3=Quality of air (days)	FR4=Price($)
A	20-40	50-70	300-320	450-550k
B	20-30	50-75	340-350	450-650k
C	25-45	50-80	350+	600-800k

Information Content Calculation

Suh, Aximatic Design, 2001

Town	I_1 [bits]	I_2 [bits]	I_3 [bits]	I_4 [bits]	Sum (I) [bits]
A	1.0	2	infinite	0	Infinite
B	0	1.32	0	0	1.32
C	2.0	1.0	0	2	5



$$I_1 = -\log_2[(30-20) / (40-20)] = -\log_2(0.5) = 1$$

Axiom 2 and Robust Design

- "The Information Axiom provides a theoretical foundation for robust design."
  - Elimination of bias
  - Reduction of Variance
    - Reduce sensitivity to variation
    - Meeting the Independence Axiom
    - Minimize random variation
    - Increase design range
  - Integrate DP in a single physical part

Comparison of Axiomatic Design with Other Methods (Suh, 2001)

- Robust design cannot be accomplished by applying the Taguchi method if the design violates **the Independence Axiom**.
- Optimization of a bad design may lead to an optimized bad design or minor improvements.
- How is Axiomatic Design similar/different from QFD?

Questions about the Axioms

- Too good to be true? What about constraints?
- Are interactions so bad? That's what makes a system great!
  - Definition of System--A combination of interacting elements organized to achieve one more stated purposes.

Lecture Outline

- ✓ Introduction to Axiomatic Design
  - ✓ Four domains
  - ✓ Axiom 1—Independence Axiom
    - ✓ Design Matrix
    - ✓ Zigzagging
    - ✓ Constraints
  - ✓ Axiom 2—Information Axiom
- ❑ Design Structure Matrix for Technical Systems
- ❑ DM—DSM Method

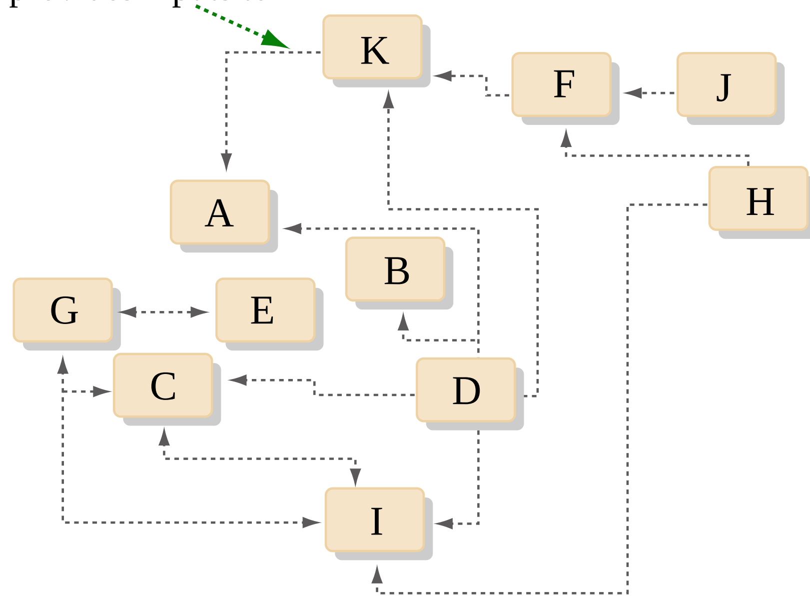
Matrix Representation of a Network
--The Design Structure Matrix (DSM)

DSM is the adjacency matrix of a network graph

	A	B	C	D	E	F	G	H	I	J	K
A				1							1
B				1							
C				1			1		1		
D											
E							1				
F								1		1	
G			1		1						
H											
I			1	1			1	1			
J											
K				1		1					

Above Diagonal Marks —— downstream task feeds information to upstream tasks. Potential rework.

K provides inputs to A

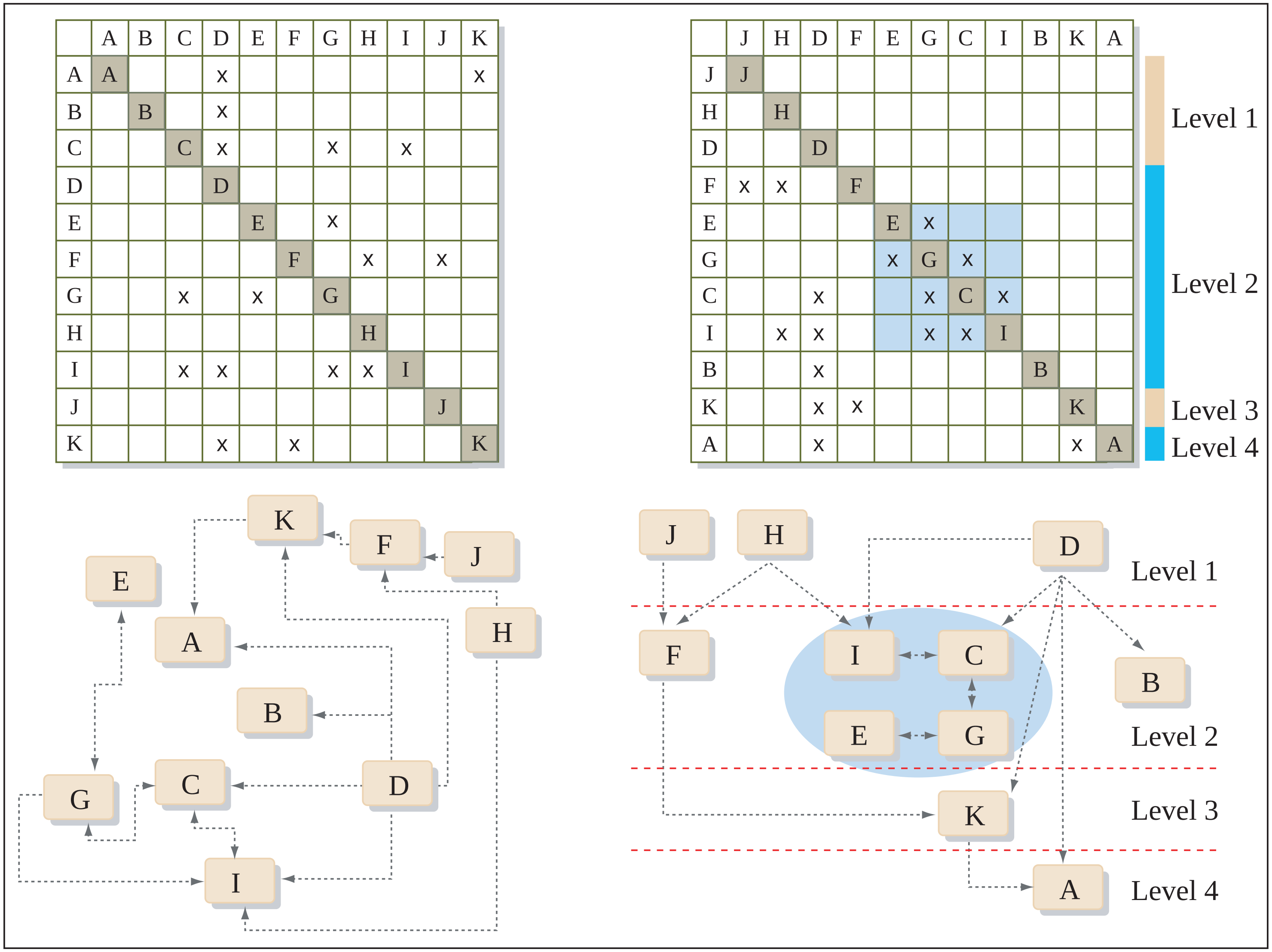


Below Diagonal Marks —— upstream task feeds information to downstream tasks

Image by MIT OpenCourseWare.

Partitioning a DSM

Before Partition / After Partition



Before Partition:

	A	B	C	D	E	F	G	H	I	J	K
A	A			x							x
B		B		x							
C			C	x			x		x		
D				D							
E					E		x				
F						F		x		x	
G			x		x		G				
H								H			
I			x	x			x	x	I		
J										J	
K				x		x					K

After Partition:

	J	H	D	F	E	G	C	I	B	K	A
J	J										
H		H									
D			D								
F	x	x		F							
E					E	x					
G					x	G	x				
C			x			x	C	x			
I		x	x			x	x	I			
B			x						B		
K			x	x						K	
A			x							x	A

Image by MIT OpenCourseWare.

Partitioning identifies truly coupled elements.

Car Door System Design

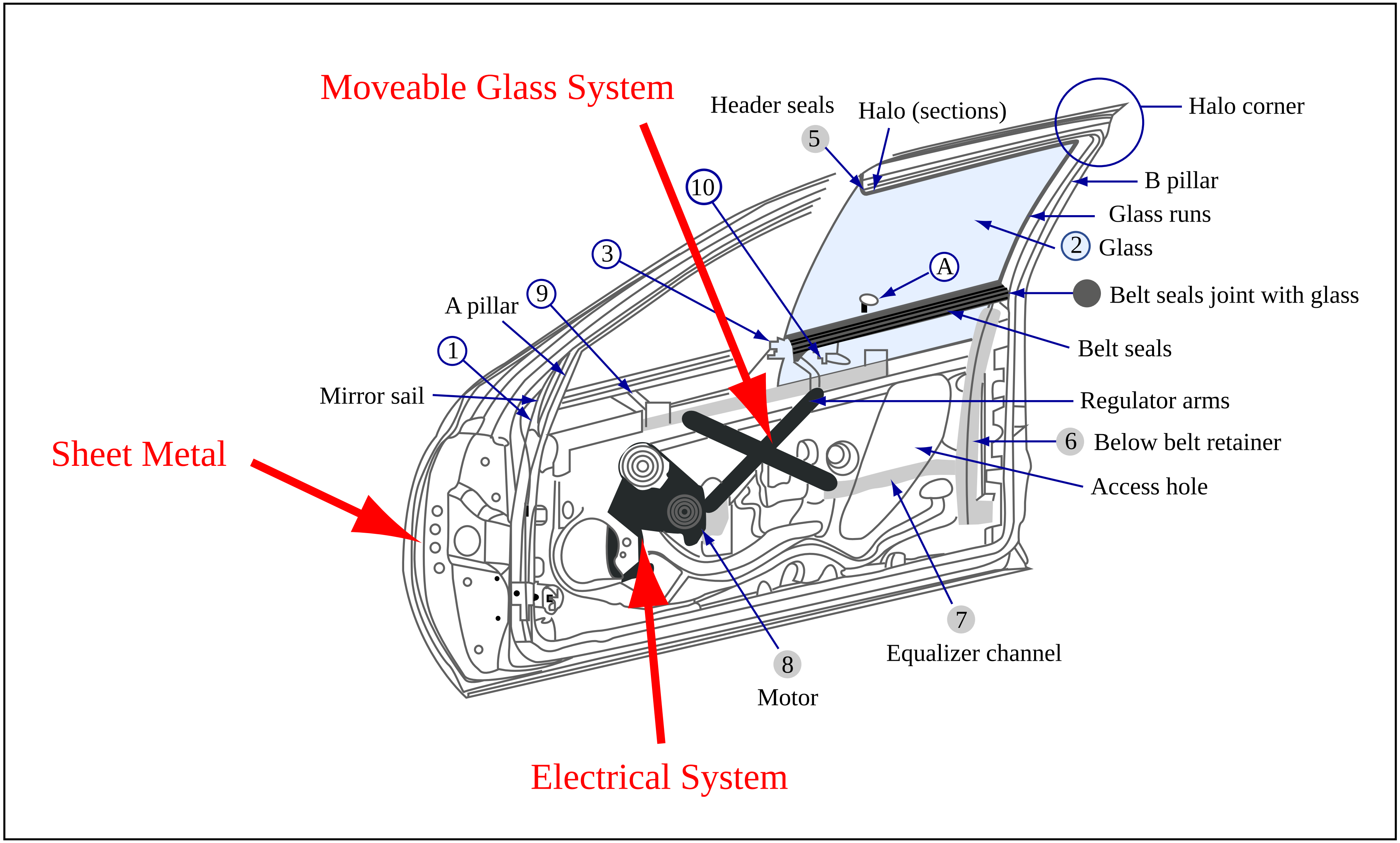
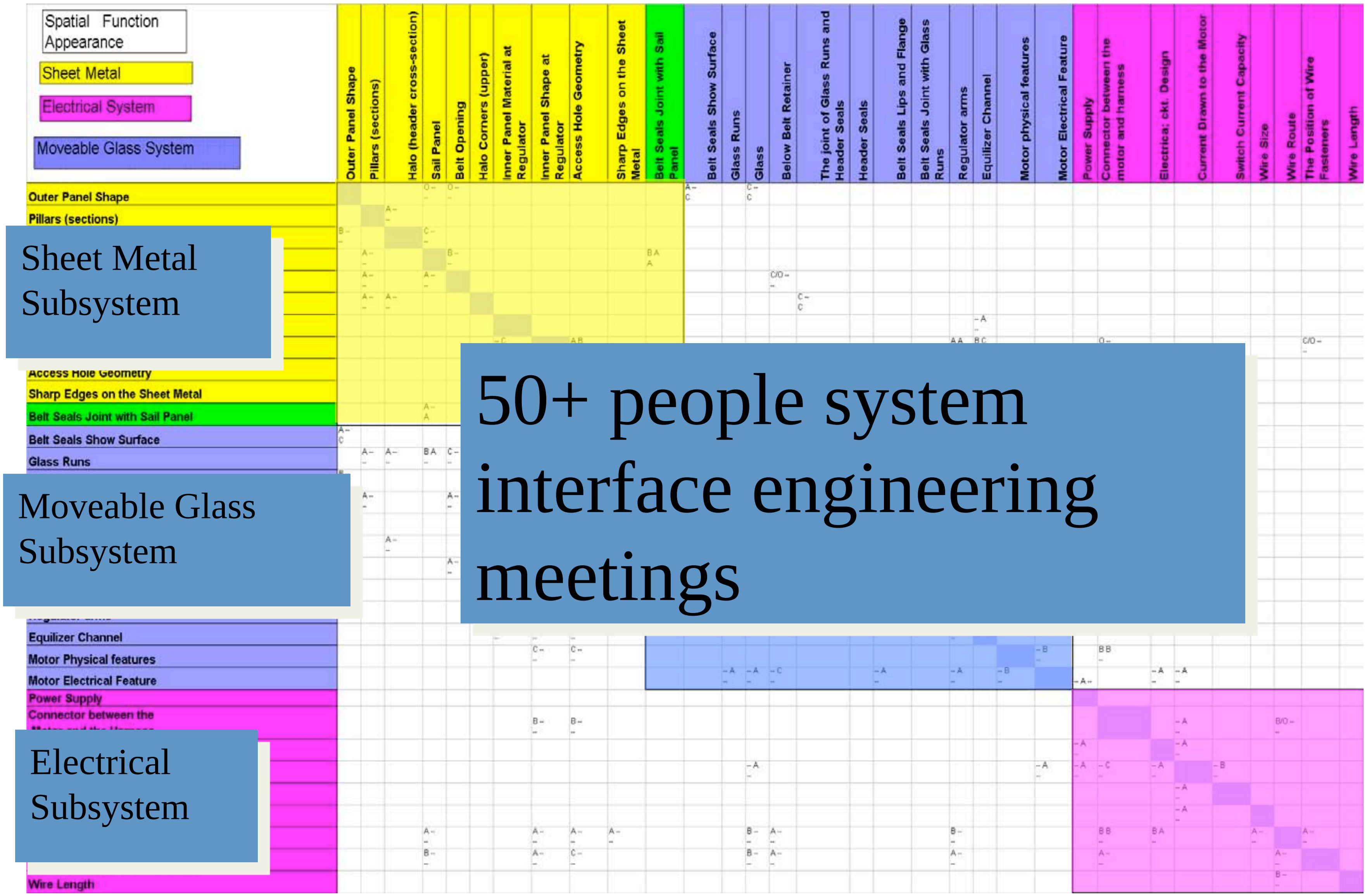
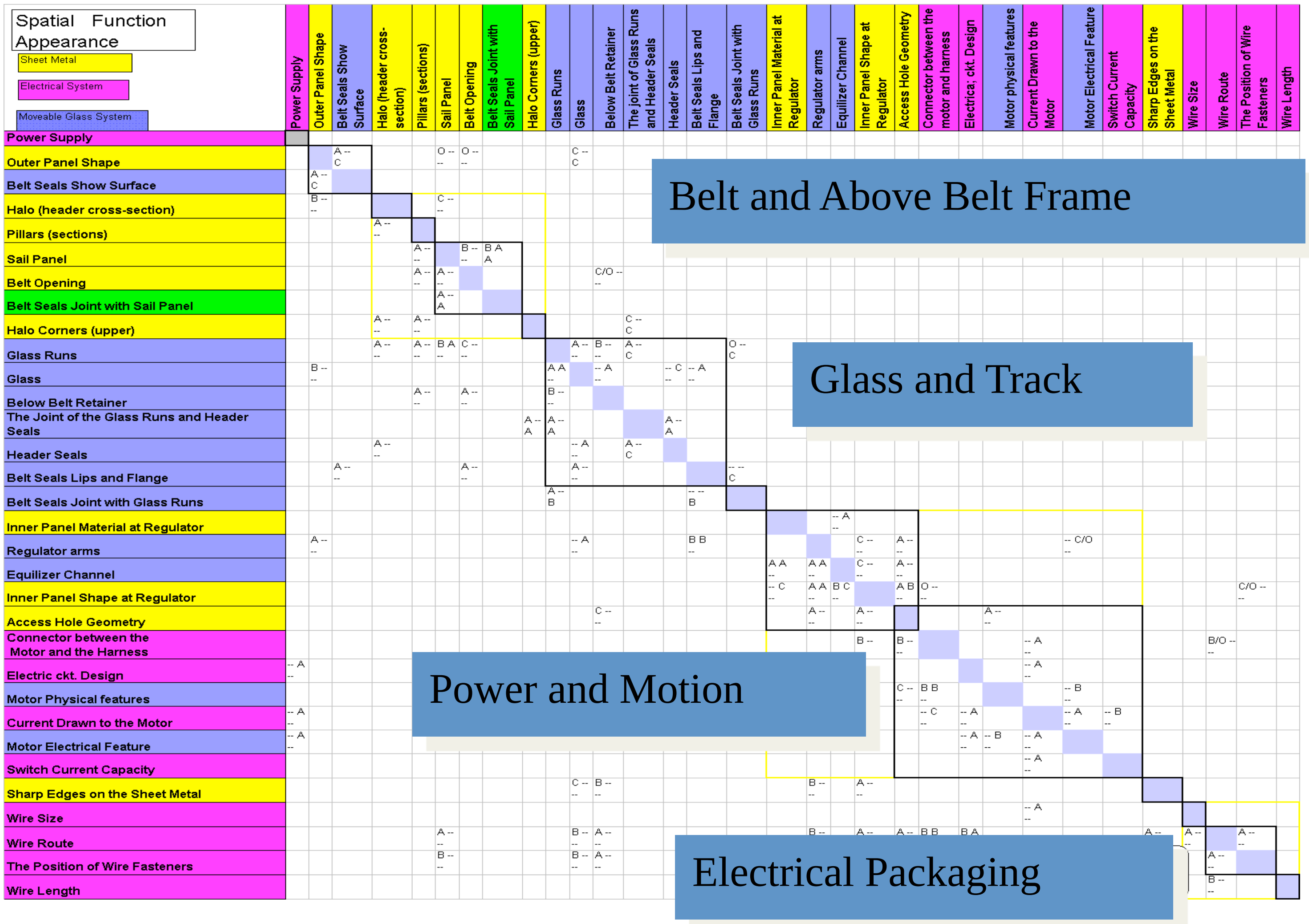


Image by MIT OpenCourseWare.

Car Door System Engineering Process (Before Partitioning DSM)

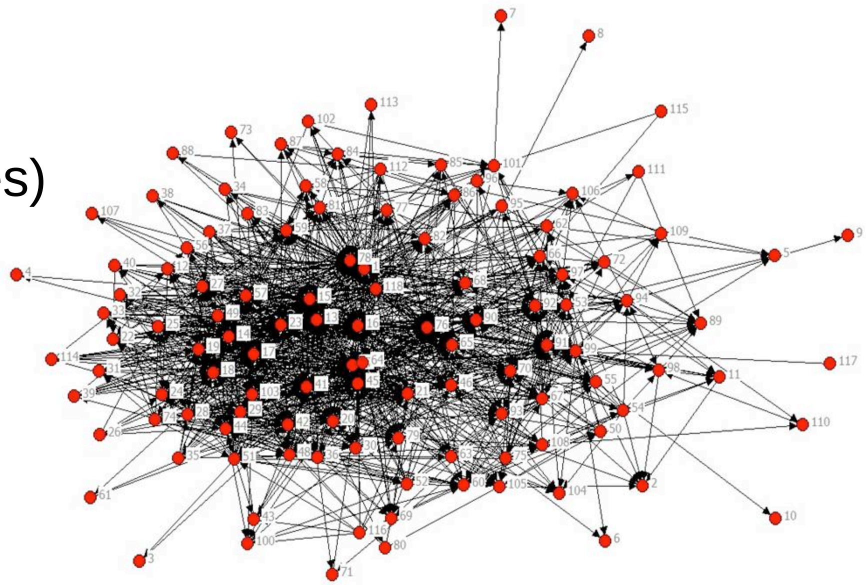


Car Door System Engineering Process (After Partitioning)



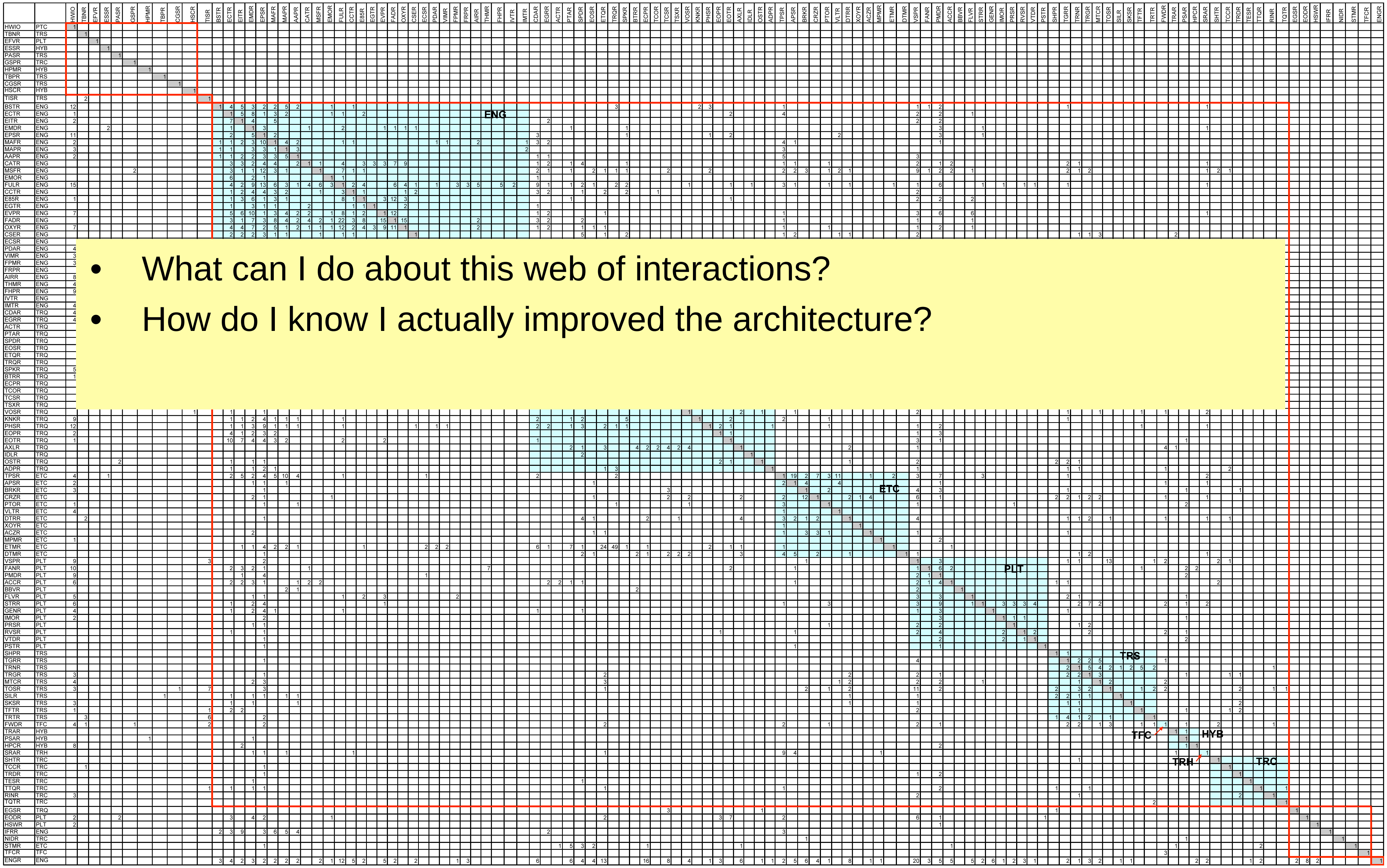
The Control Software System

- 1 production-level software
- 117 software modules (red dots)
- 1423 binary interactions (black lines)
- 39 such production software releases per year
- <2 weeks per release



- What can I do about this web of interactions?
- How can I convince management that changes are needed?
- How do I know I actually improved the architecture?

DSM of the Control System Software



- What can I do about this web of interactions?
- How do I know I actually improved the architecture?

Comparison of Various Modularity Metrics

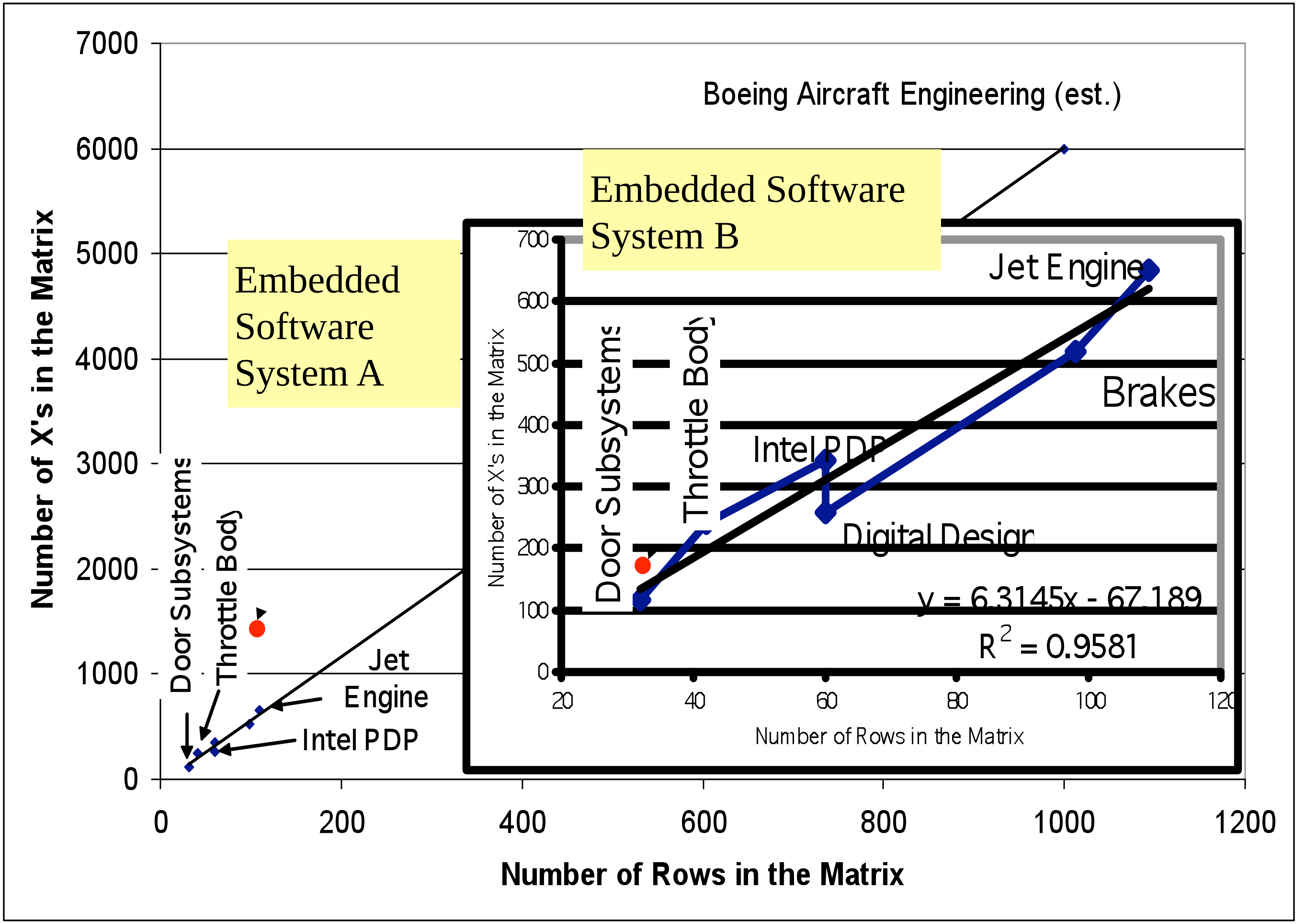
Comparison Criteria	WI	CC	SMI	VD plot	Degree Modularity	Closeness Modularity			Betweenness Modularity
						Freeman Farness	Reach Centrality	Eigenvector Centrality	
Capable of producing a consistent modularity index for the overall system	+	+							
Capable of assessing the density of immediate interactions	+								
Capable of assessing the propagation of interactions		+							
Identifies key elements in the system for modularity concerns					+	+	+	+	+
Simple to compute.	+	+	+	+					

Use Whitney Index and Change Cost to measure modularity improvements.

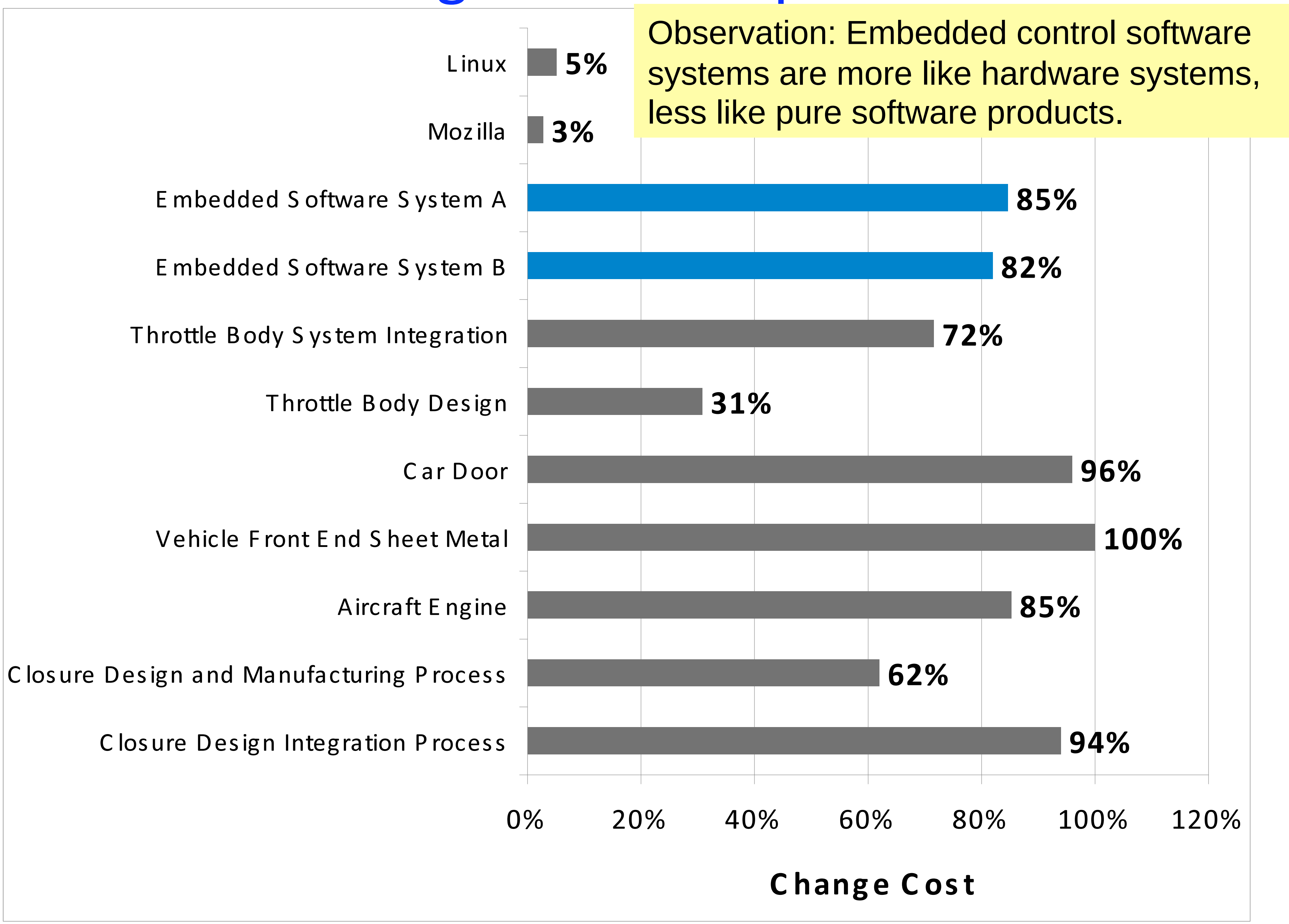
Use network centrality indices to identify system elements for improvement.

DETC2008-DTM-49140

Whitney Index Comparison



Change Cost Comparison



Network Centrality—Degree Centrality

(Sosa, Eppinger, Rowles 2007, Borgatti, Everett, and Freeman, 2002, UCINET)

In degree—how many others pass information to the element of interest.
Out degree—how many others depend on the element of interest for information.

Degree Centrality identifies which few elements, if any, in the system have a central effect on the rest of the systems.

However, the metrics values don't correlate well with components modularity.

Matrix 4

	A	B	C	D	E	F	G	H	Freeman Centrality in degree	Freeman Centrality out degree	Freeman Centrality Overall in	Freeman Centrality Overall out
A	1	1	1	1					3	3		
B	1	1	1	1					3	3		
C	1	1	1	1					3	3		
D	1	1	1	1					3	3	0%	0%
E					1	1	1	1	3	3		
F					1	1	1	1	3	3		
G					1	1	1	1	3	3		
H					1	1	1	1	3	3		

Matrix 5

	A	B	C	D	E	F	G	H	Freeman Centrality in degree	Freeman Centrality out degree	Freeman Centrality Overall in	Freeman Centrality Overall out
A	1	1	1	1	1	1	1	1	7	7		
B	1	1							1	1		
C	1		1						1	1		
D	1			1					1	1	85.70%	85.70%
E	1				1				1	1		
F	1					1			1	1		
G	1						1		1	1		
H	1							1	1	1		

	A	B	C	D	E	F	G	H	Freeman Centrality in degree	Freeman Centrality out degree	Freeman Centrality Overall in	Freeman Centrality Overall out
A	1	1						1	2	2		
B	1	1	1						2	2		
C		1	1	1					2	2		
D			1	1	1				2	2	0%	0%
E				1	1	1			2	2		
F					1	1	1		2	2		
G						1	1	1	2	2		
H	1						1	1	2	2		

Matrix 8

Image by MIT OpenCourseWare.

Network Centrality

(Sosa, Eppinger, Rowles 2007, Borgatti, Everett, and Freeman, 2002, UCINET)

- Network centrality metrics can identify the few elements that have the largest impact on the system.
- If the network has central players, the network may be bus-modular.
- If the network does not have central player, the network system is either not connected, or highly integral.
- Central players can be the priority for system complexity reduction strategy.

DSM Method

- Capture system interactions
- Analyze and improve system architecture and system interfaces.

Lecture Outline

- ✓ Introduction to Axiomatic Design
  - ✓ Four domains
  - ✓ Axiom 1—Independence Axiom
    - ✓ Design Matrix
    - ✓ Zigzagging
    - ✓ Constraints
  - ✓ Axiom 2—Information Axiom
- ✓ Design Structure Matrix for Technical Systems
- ❑ DM—DSM Method

Existing Methods Concerning System Interactions

	Design Structure Matrix (DSM)	Axiomatic Design's Design Matrix (DM)	Requirements Management	What We Want
Provide analytical system analysis	**Yes**			**Yes**
Allow iterations and feedback loops	**Yes**			**Yes**
Relate the requirements to the system design		**Yes**		**Yes**
Can be applied in the early design phases		**Yes**	**Yes**	**Yes**
Provide complete understanding of all requirements			**Yes**	**Yes**

Solving System of Linear Equations

Question: $3 * x1 + 5 * x2 = 6$ (1)

$2 * x1 - x2 = 4$ (2)

What is x1 and x2?

Solving by substitution:

Select x1 as the *output variable* in (1):

$x1 = (6 – 5 * x2) / 3$

Select x2 as the *output variable* in (2):

$x2 = 2 * x1 – 4 = 2 * (6-5*x2)/3 - 4$

$x1=2$ $x2=0$

	X1	X2
X1		x
X2	x	

Converting a DM into a DSM

1. Construct an Axiomatic Design's Design Matrix.

2. Select Output Variables.

   DP3 = *f (FR1,* ***DP1****)*
   DP1 = *f (FR2,* ***DP2****)*
   DP2 = *f (FR3,* ***DP3****)*

3. Permute the matrix by row so that the output variables are on the diagonal. We get a precedence matrix (DSM) of the Design Parameters.

	DP1	DP2	DP3
FR1	X	0	X
FR2	X	X	0
FR3	0	X	X

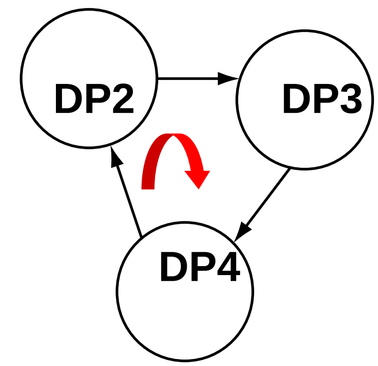
	DP1	DP2	DP3
FR1	X	0	X
FR2	X	X	0
FR3	0	X	X

	DP1	DP2	DP3
DP1	X	X	0
DP2	0	X	X
DP3	X	0	X

Selecting Output Variables

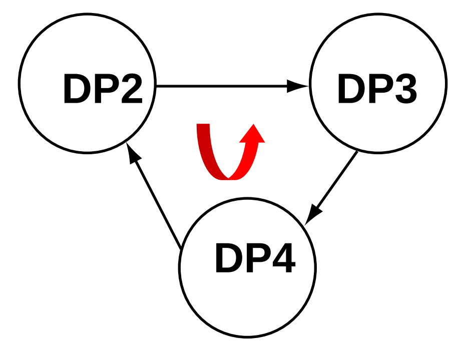
	DP1	DP2	DP3	DP4
FR1	x			
FR2	x	x		x
FR3		x	x	
FR4			x	x

	DP1	DP2	DP3	DP4
DP1	x			
DP2	x	x		x
DP3		x	x	
DP4			x	x

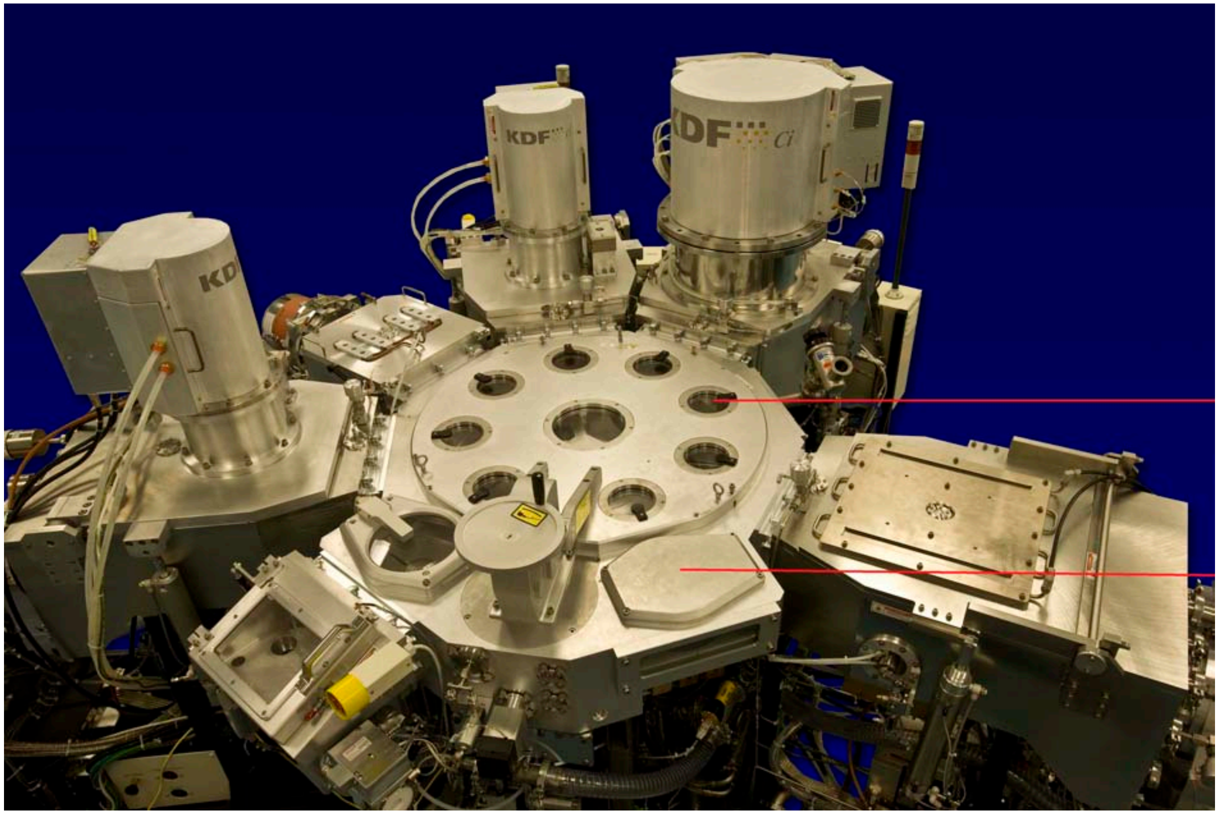


	DP1	DP2	DP3	DP4
FR1	x			
FR2	x	x		x
FR3		x	x	
FR4			x	x

	DP1	DP2	DP3	DP4
DP1	x			
DP2		x	x	
DP3			x	x
DP4	x	x		x



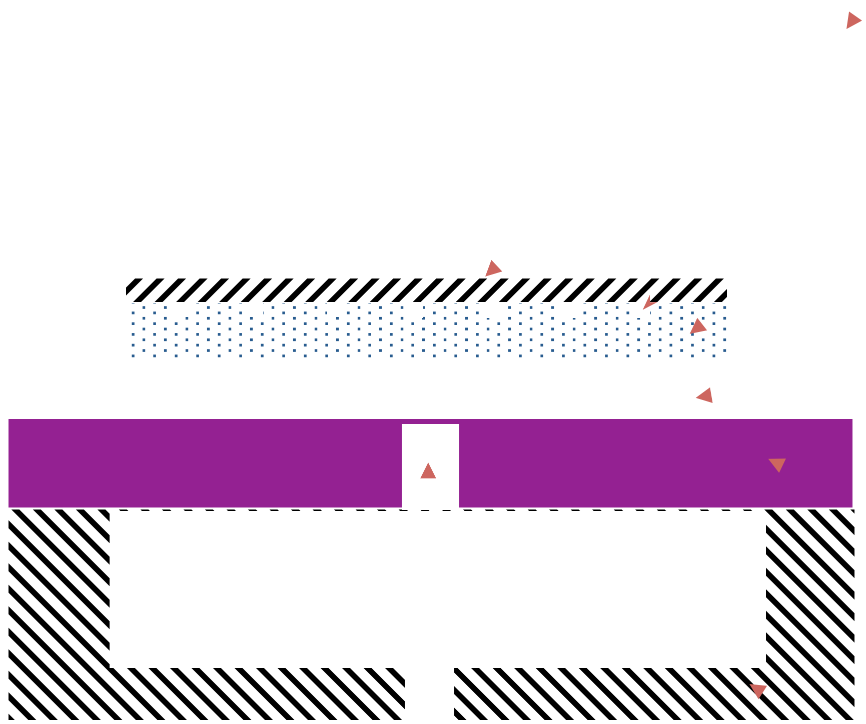
CVC Cluster Machines



Central Wafer Handler

Wafer Processing Module

CVC Electro-static Chuck (ESC)



Process Chamber

Wafer

Backside gas channel

Electro-statically charged plate

Cooling Plate

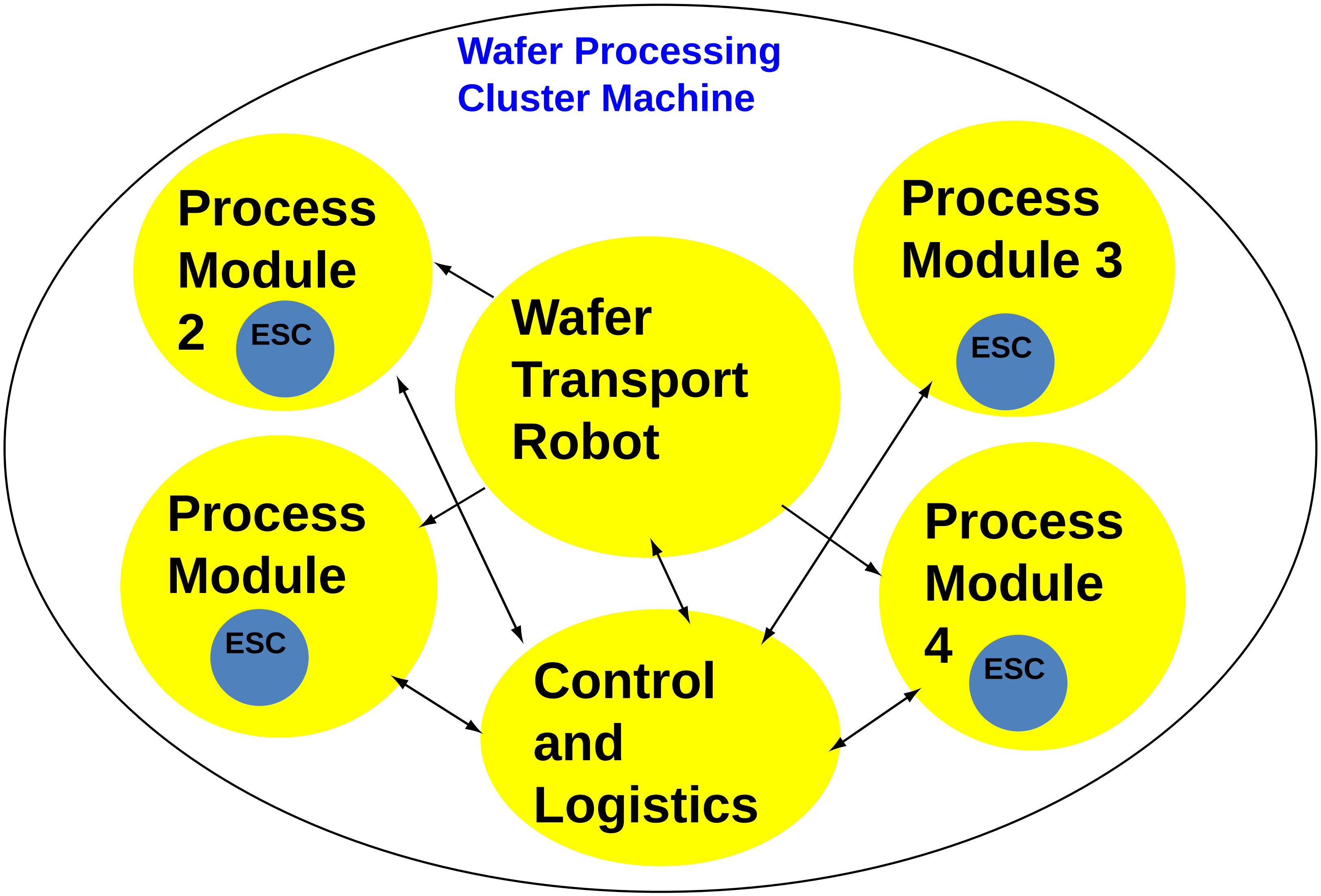
ESC

Plate for interface with various process modules

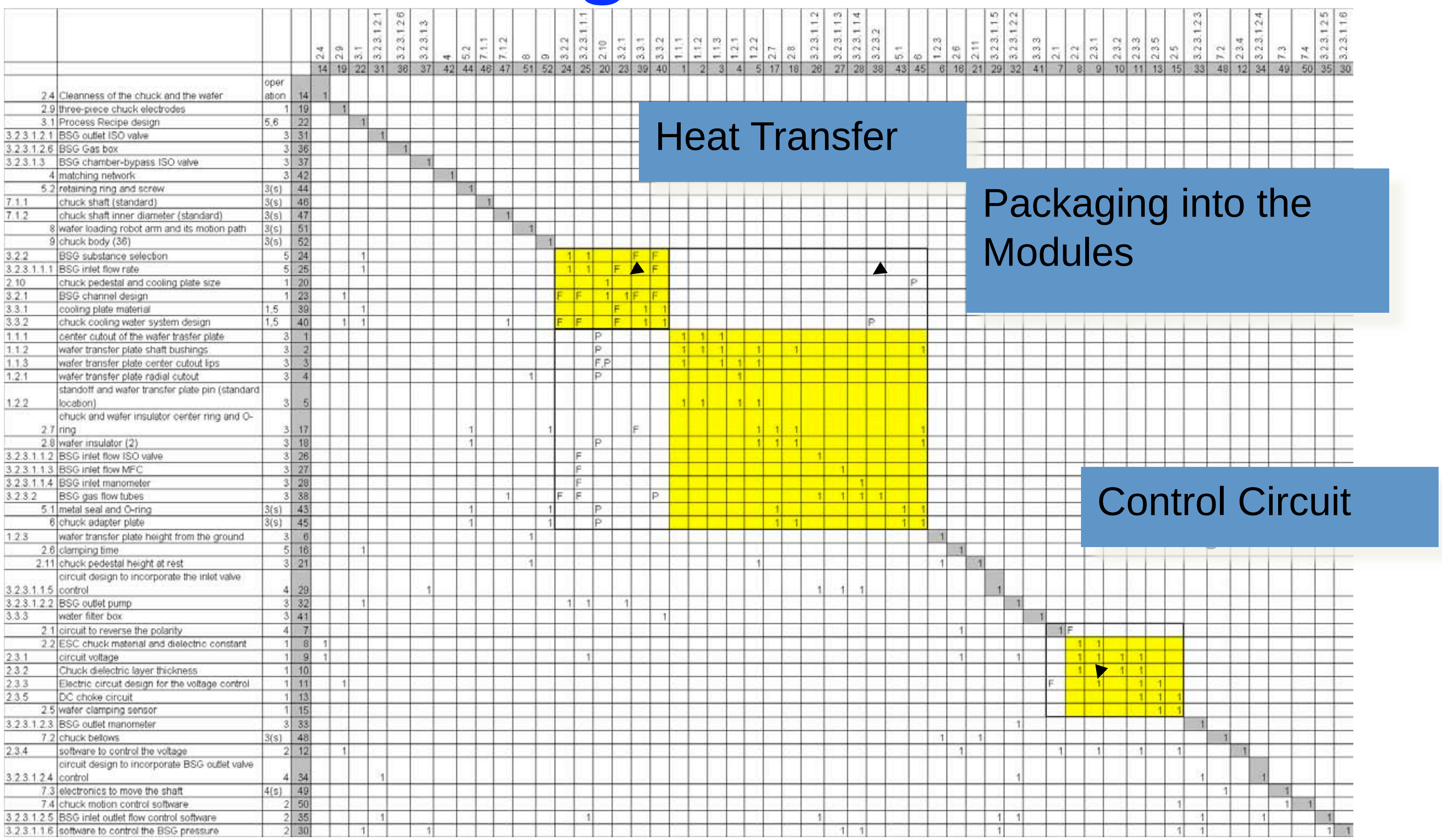
Standard interface on all process modules

Backside Gas,
Cooling Water,
Electricity

System View of ESC



Design Structure Matrix Built from Design Matrix



The Selection of Output Variables

Choosing non-diagonal elements in the DM as output variable set is like designing components not for their main functional purposes, but for their side effects. The resulting DSM is a non-executable design process.

The Selection of Output Variables

	DP1	DP2
FR1	0.75	0.2
FR2	0.2	0.9

Δ DP1 = 1/0.75 * ! FR1 - 0.2/0.75* ΔDP2
Δ DP2 = 1/0.9 * ! FR2 - 0.2/0.9 * ΔDP1

	ΔDP1	ΔDP2
ΔDP1	0	0.2/0.75
ΔDP2	0.2/0.9	0

Eigen Value = 0.243
This process converges.

	DP1	DP2
FR1	0.75	0.2
FR2	0.2	0.9

ΔDP1 = 1/0.2 * ! FR1 Ð 0.75/0.2* ΔDP2
Δ DP2 = 1/0.2 * ! FR2 - 0.9/0.2* ΔDP1

	ΔDP1	ΔDP2
ΔDP1	0	0.75/0.2
ΔDP2	0.9/0.2	0

Eigen Value = 4.1
This process does NOT converge.

The Interchangeability of DM and DSM

DM

	DP1	DP2	DP3	DP4
FR1	x			
FR2	x	x		x
FR3		x	x	
FR4			x	x

	DP1	DP2	DP3	DP4
DP1	x			
DP2	x	x		x
DP3		x	x	
DP4			x	x

DSM

The diagonal elements in the DM are the dominant elements in their corresponding rows.

Johnson and Johnson Ortho-Clinical Diagnostics OASIS Analyzer

Image of Vitros 5.1 cluster removed due to copyright restrictions.

OASIS Major Subsystems

Frame and Cabinetry (STRU)

Primary Sample Metering (SRME)

Slide Incubator (SLIN)

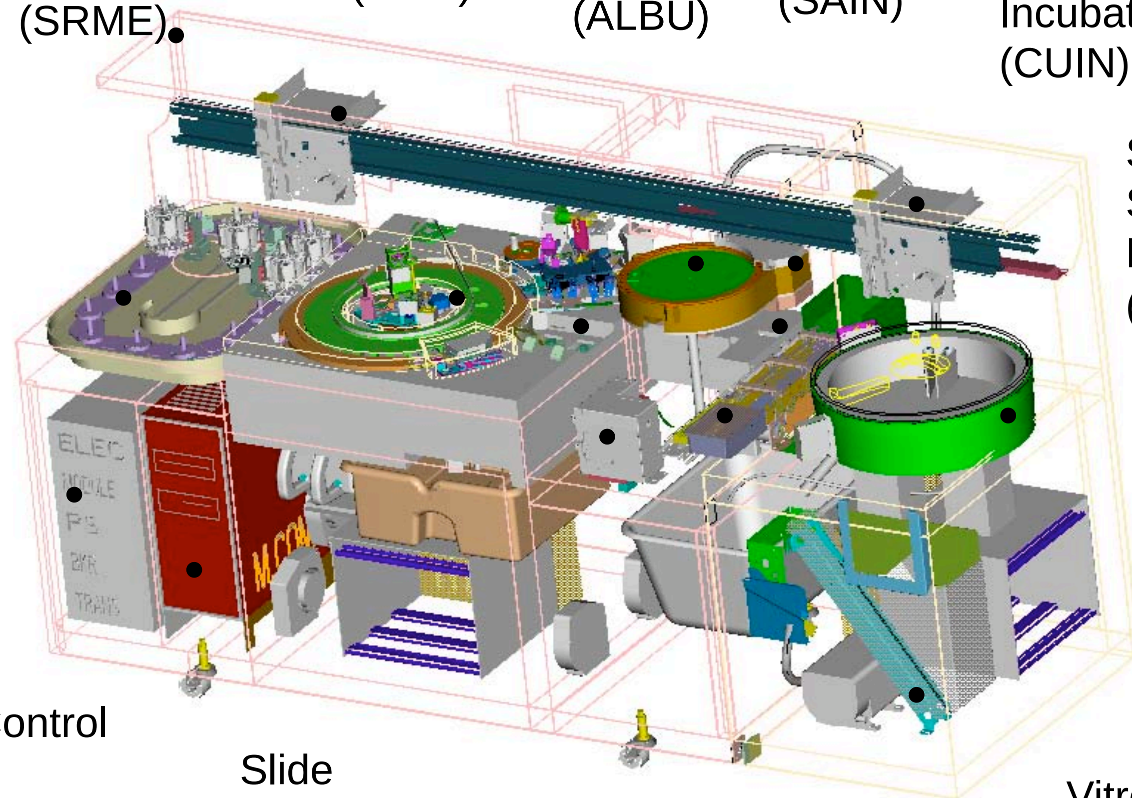
Aliquot Buffer (ALBU)

Sample Integrity (SAIN)

Cuvette Incubator (CUIN)

Sample Handling (SAHA)

Secondary Sample Metering (SRME)



Reagent Supply (RGSU)

Power Distributio n (POWR)

Machine Control (MACO) Application Services (APPS)

Slide Supply (SLSU)

Photometer (PHMT)

Microtip Loading (MTLD)

Vitros Tips Loader (VTLD)

Case Study Objectives

1. Build a DSM from requirements using the DM-DSM conversion method;
2. Compare the resulting DSM with the DSM experts built using traditional DSM construction method.
3. Understand which types of requirements can be used to predict system interactions. Judge whether the prediction DSM is complete.
4. Aid the system integration manager's work on planning and managing OASIS subsystem interfaces.

DSM Constructed from Requirements

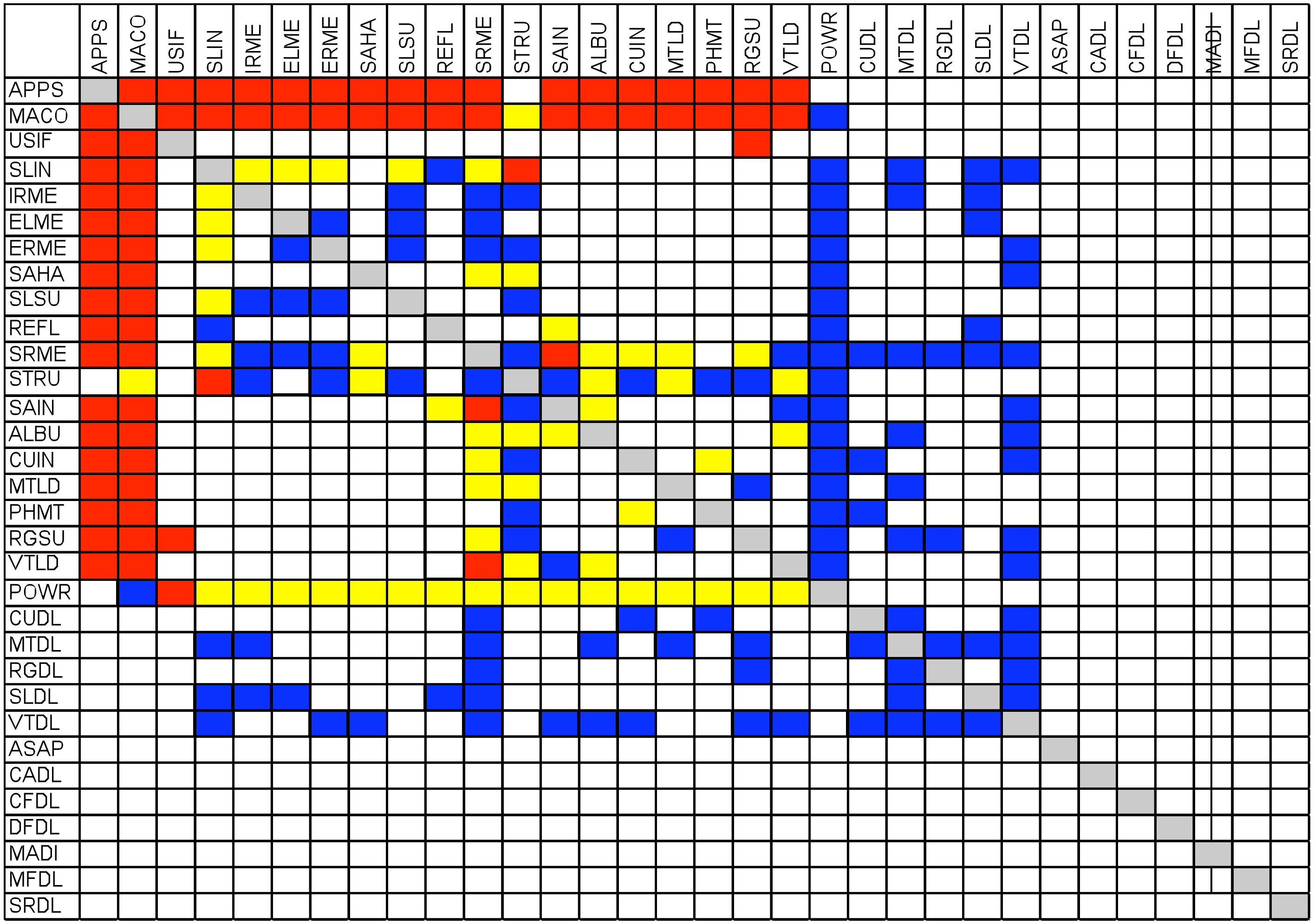
	APPS	MACO	USIF	SLIN	IRME	ELME	ERME	SAHA	SLSU	REFL	SRME	STRU	SAIN	ALBU	CUIN	MTLD	PHMT	RGSU	VTLD	POWR	ASAP	CADL	CFDL	CUDL	DFDL	MADI	MFDL	MTDL	RGDL	SLDL	SRDL	VTDL
APPS		X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X	X													
MACO	X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X													
USIF	X	X																X														
SLIN	X	X			X	X	X		X		X	X																				
IRME	X	X		X																												
ELME	X	X		X																												
ERME	X	X		X																												
SAHA	X	X									X	X																				
SLSU	X	X		X																												
REFL	X	X											X																			
SRME	X	X		X				X					X	X	X	X		X	X													
STRU		X		X				X						X		X			X													
SAIN	X	X								X	X			X																		
ALBU	X	X									X	X	X						X													
CUIN	X	X									X						X															
MTLD	X	X									X	X																				
PHMT	X	X													X																	
RGSU	X	X	X								X																					
VTLD	X	X										X		X																		
POWR			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X													
ASAP																																
CADL																																
CFDL																																
CUDL																																
DFDL																																
MADI																																
MFDL																																
MTDL																																
RGDL																																
SLDL																																
SRDL																																
VTDL																																

Software

Thin Film

Wet Chemistry

Compare Requirements DSM with Expert DSM



System Interactions Predicted by DSM from requirements.

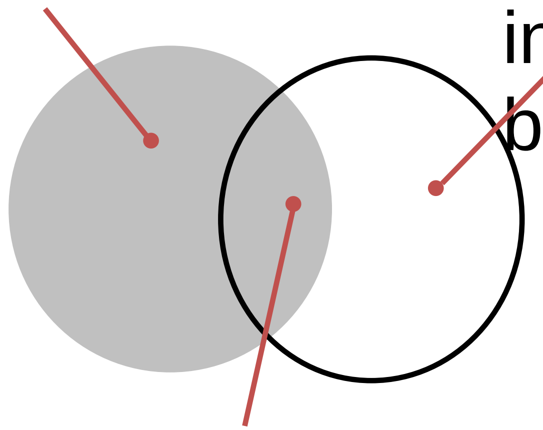
System Interactions Predicted by JNJ engineers.

System Interactions Predicted by both the requirements and JNJ engineers.

How Many Marks Match

The experts did not capture 75 interfaces predicted by the requirements.

The requirements prediction DSM missed 118 interactions captured by the experts.



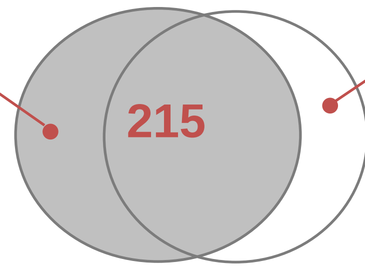
There are 54 marks captured by both the experts DSM and the DSM from requirements.

Analyzing the Unmatched Marks

Type of Missing Mark	Number of missing marks	Reason for Missing	Who Missed them	Remedy	Problem of the matrix conversion method
(1) Hardware-software interaction	69	The experts did not involve software people in the DSM exercises.	JNJ engineers	involve software people in the next DSM building exercise	no
(2) Assay-hardware interaction	64	No assay design requirement has been documented.	requirements DSM	JNJ chemists produce assay design requirements documents.	no
(3) Power subsystem interaction	17	The power subsystem engineer says there will be no need for information feedback to other subsystems.	requirements DSM	Does not count as a mistake.	no
(4) Reliability induced interaction	12	Reliablity requirement decomposition is difficult to use to predict system level tradeoffs.	requirements DSM	Use past design history on relialbity issues (e.g. the hazard analysis document at JNJ)	**yes**
(5-1) Function types of interaction	11	Not reflected in reuqirements decomposition structure.	requirements DSM	better requirements writing and management	no
(5-2) Spatial types of interaction	14	Spatial relationship is not detailed by requirements.	requirements DSM	Use Datum Flow Chain	**yes**
(6) experts missed interaction	6	experts did not bring them up during DSM building exercises.	JNJ engineers	JNJ engineers can learn from the requirements driven DSM.	no

The Achievable Potential

JNJ experts miss 6 marks.



DSM from requirements misses 26 marks.

Providing:

- JNJ engineers involve software engineers in the DSM building exercise
- Chemists write assay requirements
- JNJ updates the trace-ability between product level requirements and subsystem level requirements

Limitation of the Method

Can all requirements be decomposed to predict system interactions?

Requirements Decomposition

	Can predict system interactions	Cannot predict system interactions
Can be decomposed in the same way as the FR's in the Axiomatic Design	Functional, Maintainability, Operational, Environment Expandability, Appearance	None
Can be decomposed but not in the same way as decomposing the FR's in the Axiomatic Design	Performance (Modeling) Packaging (DFC, DSM) Design Constraints (DSM)	Reliability (budgeting) Size (budgeting) Weight (budgeting) Cost (budgeting)
Difficult to decompose	None	Installation Standards Safety DFMAS Component Reuse Operability Shipping
No strong evidence in this case study	Disposal Distribution Training Budget and Timing Patents	

.

Comparison of the Three Methods

Axiomatic Design Matrix (DM)

X	
	X

X	
X	X

Uncoupled Design

Decoupled Design

Avoids coupling by smart engineering design.

DSM

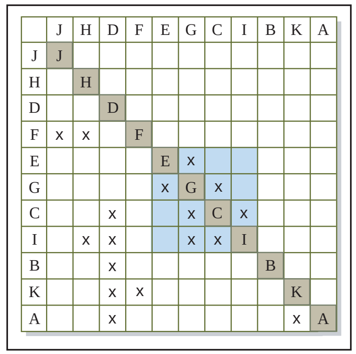


Image by MIT OpenCourseWare.

Accepts coupling and manage it by streamlining the process, or modularizing the system architecture.

DM - DSM

- Reduce the amount of coupling through good design.
- Manage the inevitable coupling when a coupled design makes more business sense.
- DSM shows the bottleneck in systems and ultimately drive people toward Axiomatic Design preferred results.

Summary of DM-DSM Method

- We can get a DSM from a DM
- The diagonal elements are the output variables in matrix conversion
- Not all system interactions can be predicted from DM
- Coupled design can be managed and improved using DSM.
- Do think about reducing system coupling by exploring alternative design concepts first.

Lecture Summary

- ✓ Introduction to Axiomatic Design
  - ✓ Four domains
  - ✓ Axiom 1—Independence Axiom
    - ✓ Design Matrix
    - ✓ Zigzagging
  - ✓ Axiom 2—Information Axiom
- ✓ Design Structure Matrix for Technical Systems
- ✓ DM—DSM Method