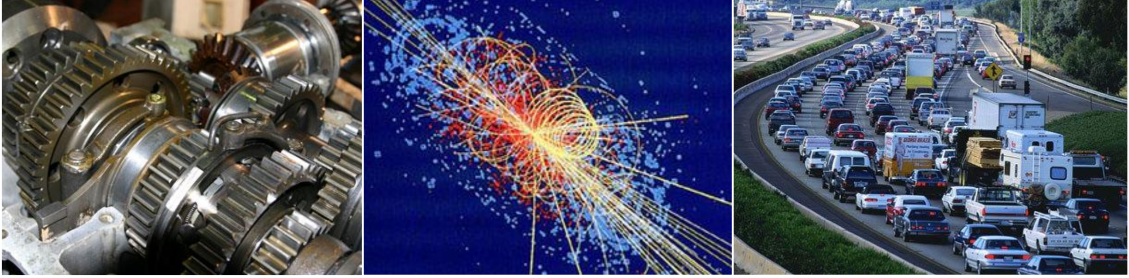
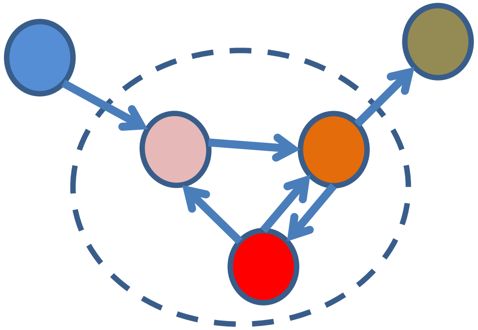




System Interactions

Making the Heart of Systems More Visible

William D. Schindel

ICTT System Sciences schindel@ictt.com

Crossroads of America Chapter

GLRC 2013: ***Leadership Through Systems Engineering***

Abstract

- This presentation argues that the most fundamental concept of systems receives less explicit attention than deserved in typical approaches to systems projects and life cycles. That fundamental concept is the notion of System Interactions. While not completely invisible in typical systems engineering processes, Interactions frequently seem to lurk just below the surface of system representations and engineering deliverables. This is true in Model-Based Systems Engineering (MBSE) as well as its predecessors. The cost of this “fog” is both missed opportunities and unpredicted problems or surprises.
- By making Interactions the explicit heart of systems representations, the author has observed dramatic improvement in the ability of individuals and teams to analyze, understand, and communicate critical systems information. This approach has been verified across domains including mil/aerospace, automotive, construction, telecommunications, medical/health care, advanced manufacturing, and consumer products. In addition, it firms up the scientific basis for systems engineering, because the physical laws of the hard sciences are virtually all statements about physical interactions.
- This presentation is for systems planners, engineering practitioners, system thinkers, and leaders. It includes a review of commonplace System Interactions in real systems, how they appear and don’t appear in typical engineering representations, and the practical impacts of this gap. It also includes examples of how this can be addressed within typical engineering and life cycle processes. The result is improved understanding, earlier awareness, and better project and life cycle performance.

Contents

- Motivating challenges
- A different view emerging from MBSE / PBSE
- The fundamental definition of "System"
- Lessons from 300 years of science
- Example uses of this approach
- Implications: What you can do

- References

Motivating Challenge:

1. Discovering System Requirements

- Traditional historical view:
  - Requirements are prose statements whose structure, grammar, and clarity are important (atomic, objective, testable, etc.)
  - Requirements are about “what”, not “how”
  - “Functional” requirements are only a subset of all the requirements for a system
  - Requirements are derived from understanding of stakeholder needs
- These are all valid in context, but the typical experience under this approach includes the following sub-optimal outcomes:
  - Different people reading the same requirements document have different interpretations of what it says
  - Evaluating the completeness and consistency of a requirements document depends heavily on subject matter experts, and subjective judgments
  - Overlooked requirements are sometimes discovered later than we’d like
  - Design constraints are sometimes mistaken as requirements

Motivating Challenge:

2. Unexpected system problems caused by environmental systems, including humans

- Traditional historical view:
  - I am responsible for my system's design—someone else is responsible for the design of other nearby systems
  - We document the expected roles of human users/maintainers after the design is far enough along
- These are likewise valid in context, but typical experience includes:
  - An external system caused unexpected problems for the system I designed
  - Human users/operators/maintainers lack of awareness or skills are causing problems for my system

Motivating Challenge:
3. Understanding causes of emergent behavior

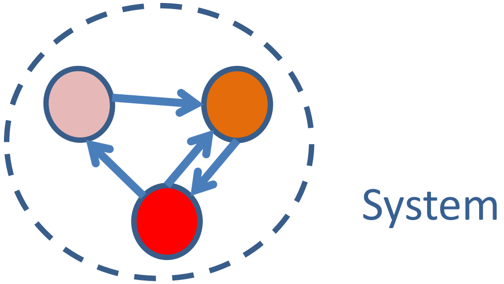
- Traditional historical view:
  - System designs (the “how”) should satisfy system requirements (the “what”)
  - The satisfaction of requirements is verified by a combination of design review and testing (including simulation)
  - Subject matter expertise / experience with the technology is central to assessing whether a design is likely to conform to requirements
- These are likewise valid in context, but typical experience includes:
  - Systems exhibit some behaviors that were not expected, in functionality, performance, usability, reliability, etc.
  - System components interact with each other in unexpected ways, with emergent consequences discovered late
  - Changes to existing designs cause unexpected negative consequences that are discovered later than we’d prefer

A different view, emerging from MBSE / PBSE

- The rise of model-based SE methods adds to the arsenal we have for addressing those concerns.
- Converting to a model-based representation may help, but does not by itself assure an answer:
  - But, a model-based approach that is based on explicit interaction models for requirements and design can make a tremendous difference.
  - Repeating, holistic Patterns of these are likewise clearer.
  - A note of caution: Not all model-based approaches necessarily include the following discipline.

The Fundamental Definition of “System”

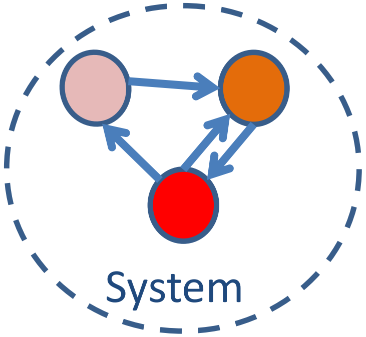
- Definition: A <u>system</u> is a set of interacting components:



  - By “interact”, we mean one component changes the state of another, through physical exchange of energy, force, mass, or information. (The last of these is really a case of the first three.)
  - By “state” of a component, we mean a property of the component in time that influences its behavior in future interactions.
  - Notice the circularity of these definitions (relational model).

Compared to a traditional view

- Two different starting points for defining System:

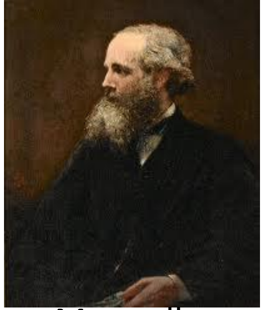


(a)

(b)

- What seems like a small difference of perspective turns out to be fundamental to thinking about systems.

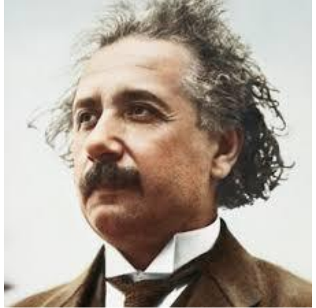

Lagrange

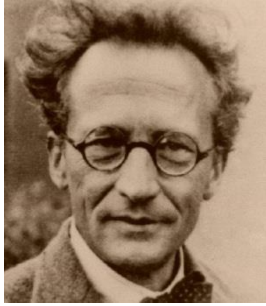

Maxwell


Mach

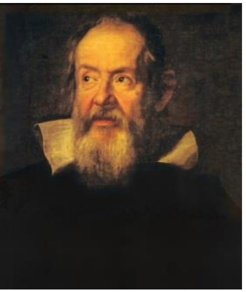

Boltzmann


Planck


Einstein


Schrodinger

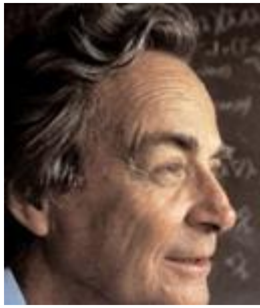

Newton


Galileo

Interactions:
Lessons from 300 years of science

- The physical laws discovered by science are all descriptions of interactions!
- Each engineering discipline (ME, EE, ChE, etc.) is based upon those scientific laws.
- We ask the same of Systems Science, as a basis for Systems Engineering.
- *Bringing Interactions front and center in Systems Engineering improves our capabilities.*


Bohr

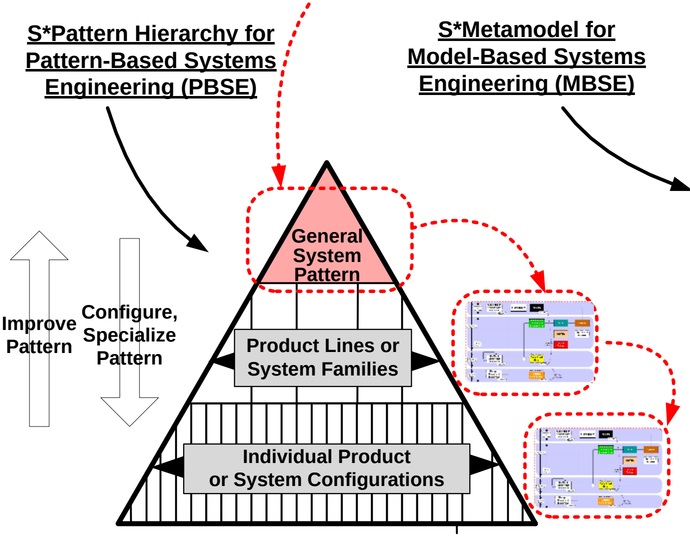

Feynman

Connections to MBSE, PBSE

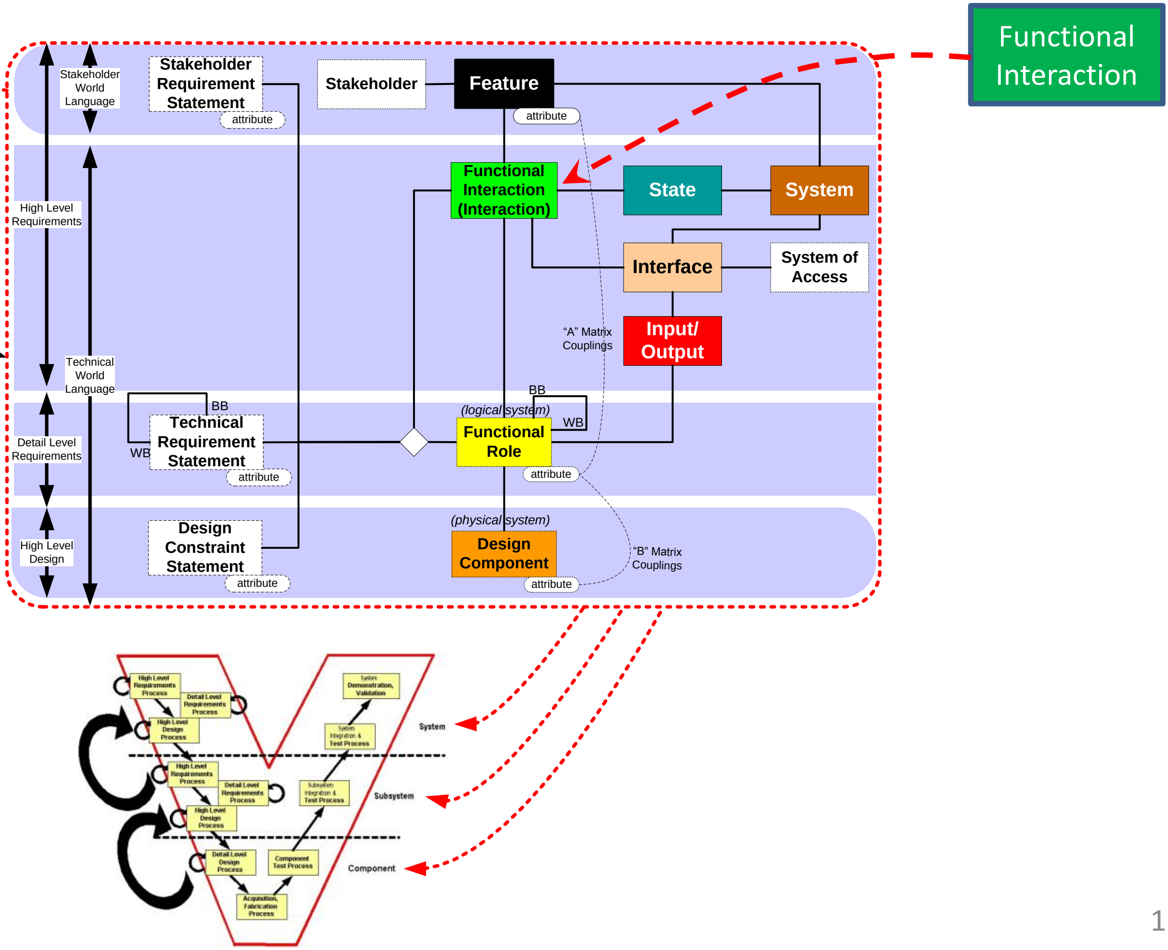
- The rise of Model-Based Systems Engineering (MBSE) offers a great opportunity to make physical interactions more explicit:
  - Indeed, these interactions can be glimpsed surfacing in Interaction Diagrams of various types, in SysML and other languages;
  - However, Interactions are not necessarily identified as fundamental objects in these models, missing an opportunity;
  - By treating Interactions as fundamental classes, along with their relationships to other classes likewise fundamental, new insights and payoffs follow;
  - Although SysML and other modeling languages allow us to make Interactions explicit, they don't necessarily force it to happen.
  - What is the smallest model of a system necessary for practical engineering?

Connections to MBSE, PBSE

- That is why the (language independent) S*Metamodel, explicitly emphasizes <u>Functional Interaction</u> as a fundamentally coordinating class relating other information—and it is readily included in SysML or other models.
- PBSE is likewise explicit in its representation of Patterns of Interactions, as would be the case in the physical sciences.



Pattern Class Hierarchy



System Containment Hierarchy

[Ref: "What Is the Smallest Model of a System?", IS2011]

Example uses of this approach

1. System requirements discovery
2. Unexpected system problems caused by environmental systems, including humans
3. Understanding causes of emergent behaviors

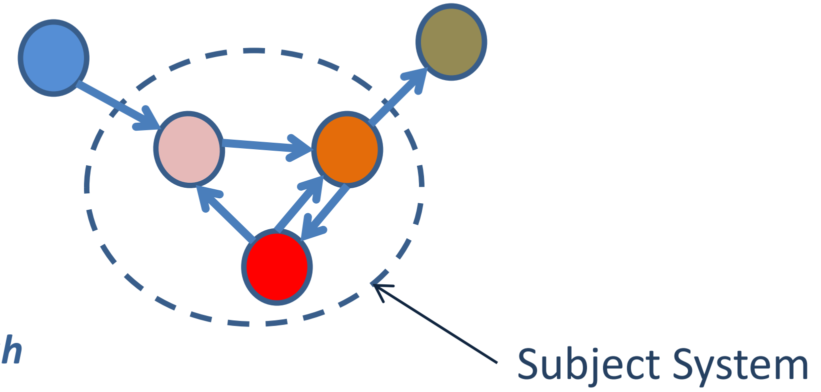
1. Discovering System Requirements

(We are referring here to the technical system requirements consumed by design & verification processes—not stakeholder requirements, from which they might have been derived.)

- Interactions-based view:
  - All system Requirements are descriptions of external system Interactions
  - System Requirements may be discovered by discovering Interactions, which are easier to find in a systematic way
  - Requirements statements (even though in prose and subject to typical requirements writing stylistic advice) take on a new interpretation that makes it easier to think about the content of each statement—including spotting design constraints and separating them
  - Requirements have a more objective basis for understanding in a common way across those who read them
  - Evaluating the completeness and consistency of Interaction-based Requirements is more practical to do in a systematic way
- Let's see why this is so . . .

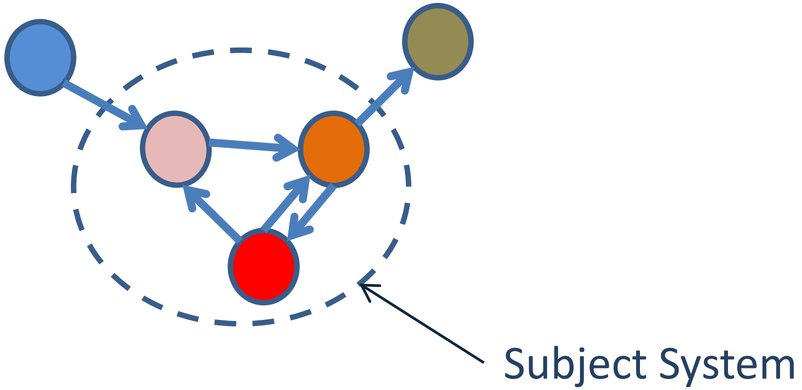
“Black Box” System Requirements

- All system Requirements are descriptions of <u>external system Interactions</u>:
  - System Requirements can describe only behavior.
  - Only external behavior is of interest: “Black Box” Requirements (see case [3] later below for “White Box” Requirements generated in design).
  - The only way that behavior can be external is as Interactions with external actors.
  - In the universe we know, those Interactions can only be exchange of energy, force, mass, or information (the latter being carried by the first three).
  - For the Subject System boundary below, where does that behavior appear?



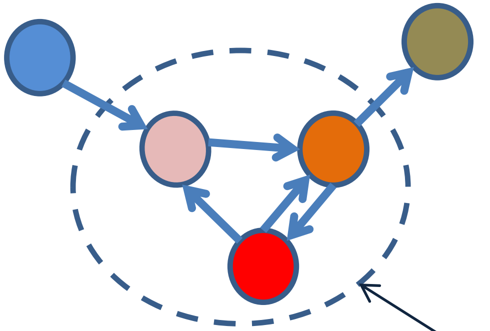
Reminder: Why are “requirements” treated that way?

- For very practical “commercial” reasons:
  - Requirements tell us what a “replacement” system would have to do, if we were to “drop it in place”, to replace a previous successful system.
  - Likewise, what a candidate design must externally satisfy.
  - That is, in a practical sense, they would be indistinguishable from the outside—or if different in behavior, then sources of possible impacts.
  - This is the eminently pragmatic, commercial reason for this approach.
  - And, it leaves us free to separately describe other issues:
    - The value / utility landscape of stakeholders, placing value on that external behavior
    - The internal design of the system, producing that external behavior.



Reminder: Why are “requirements” treated that way?

- So:
  - Requirements need only (and can only) tell us the Input-Output Behaviors of our system
  - How the Outputs are “related” to the Inputs, as to quantity, time, or other parameterized aspects
  - Requirements as transformations of Inputs to Outputs



Subject System

[Ref: “Requirements Statements Are Transfer Functions”, IS2005]

So, if we can find all the Interactions, we can find all the Requirements!

- This turns out to be a very powerful insight, because it moves the problem to something we can do very systematically.
- External Interactions may be sought out in three contexts:
  - **Domain Actor /Interface Trace**: Trace through the external interfaces / external actors, seeking out the interactions of the subject system with each of those actors. This tells us "who" or "what" the system interacts with externally.
  - **State / Mode Trace**: Trace through the states / modes / situations of the subject system, seeking out the interactions that occur during each state / mode / situation. This tells us "when" the system interacts externally. (*)
  - **Stakeholder Feature Trace**: Trace through the stakeholder features / value packages of the subject system, seeking out the interactions that directly deliver each feature / value package. This tells us "why" (in a value sense) the system interacts as it does.

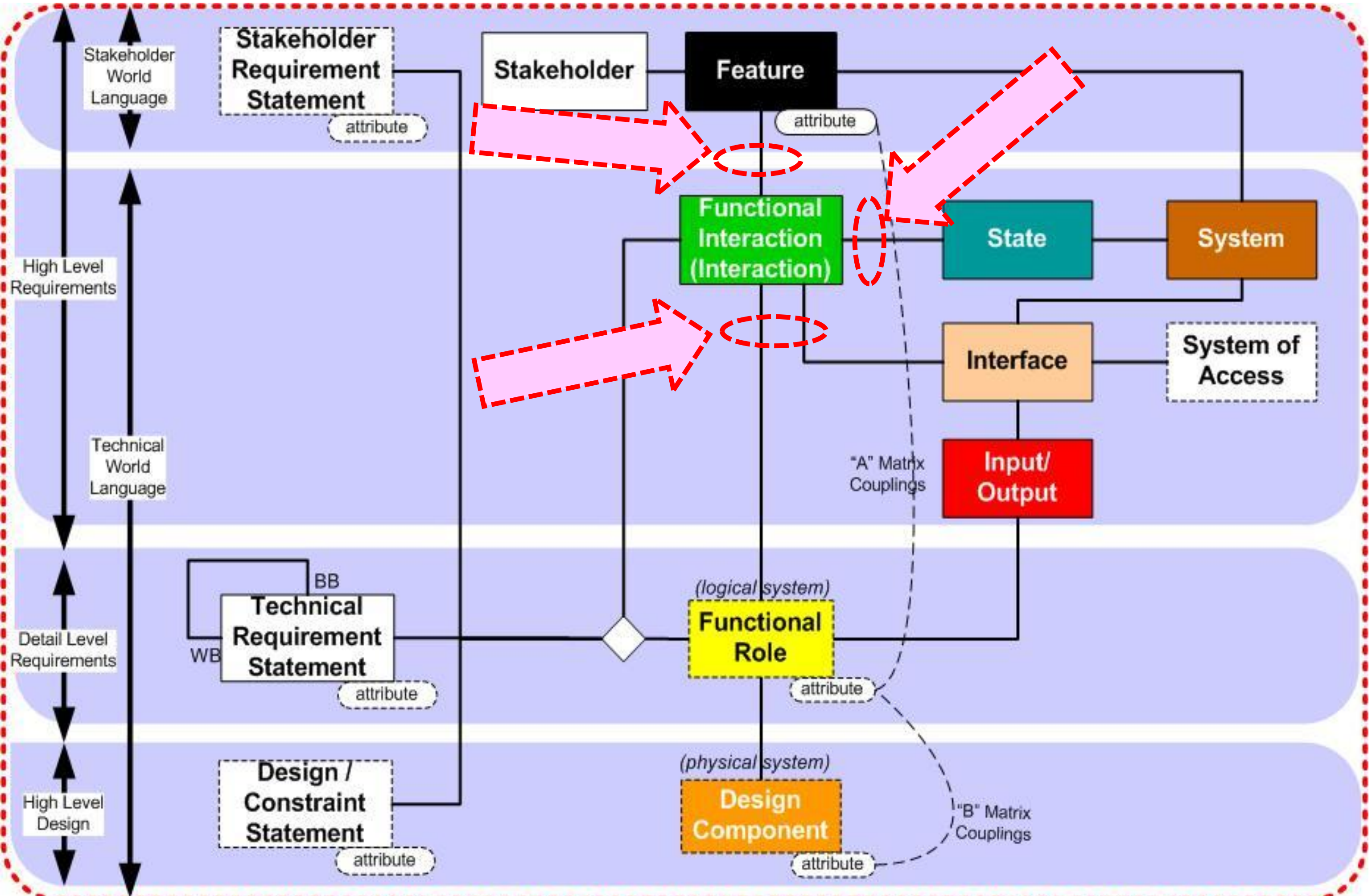
(*) The State / Mode Trace cross section is akin to the structure of Use Cases (i.e., situational).

How to find all the Interactions?

- Each of those three contexts should produce exactly the same list of Interactions:
  - However, it is typically more likely to discover some of the Interactions first in one of these contexts, thereafter locating it in the other contexts.
  - This builds a more complete set of Interactions—and therefore Requirements.

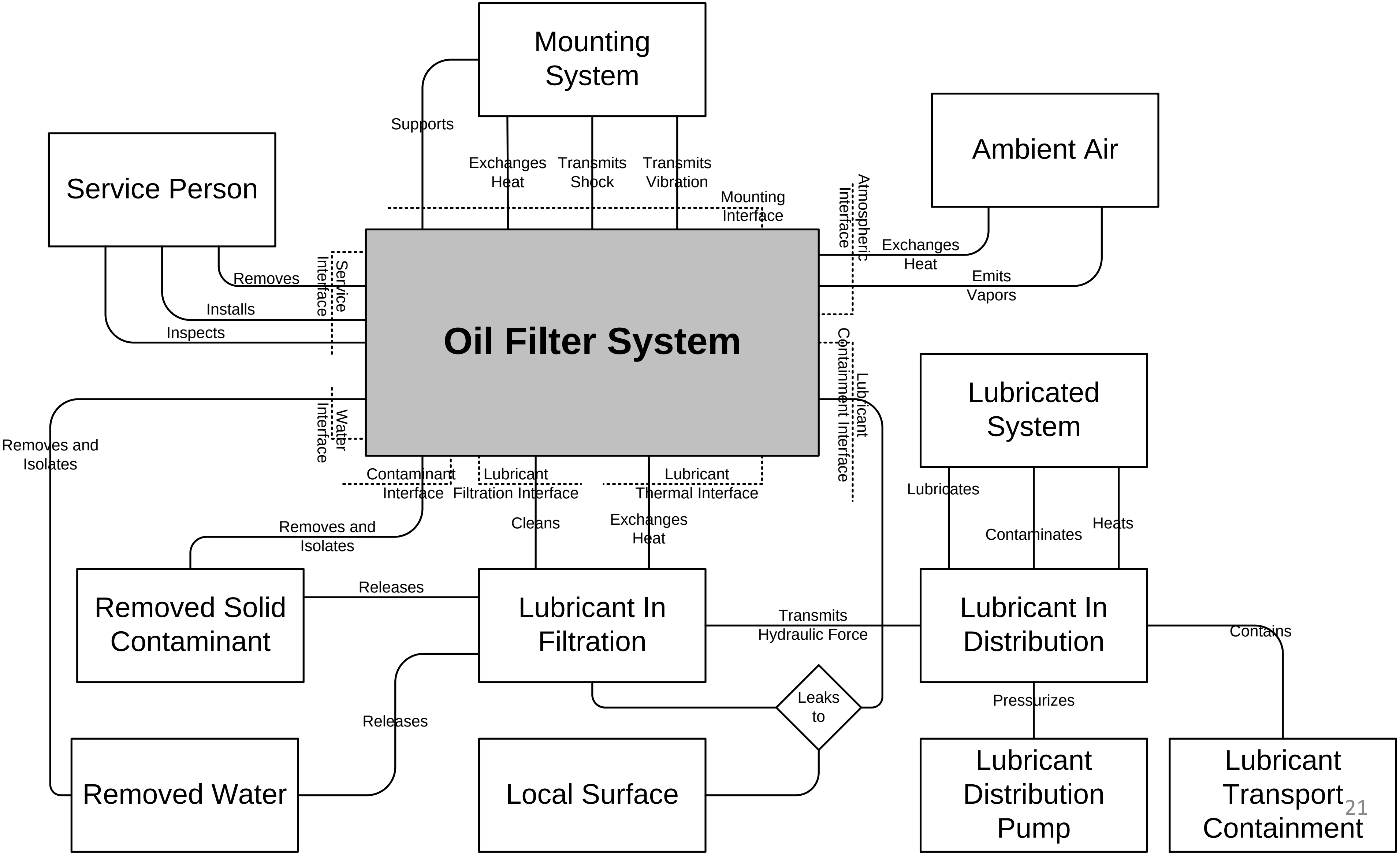
How to find all the Interactions?

- The three traces to Interactions, seen in the S*Metamodel:



A Simple Example: Oil Filter

- Domain Model shows all external actors/interfaces, over life cycle:



A Simple Example: Oil Filter

Interactions Model (explicit objects)

Store Packaged Product

Transport Packaged Product

Identify Packaged Product

Display Packaged Product

Purchase Packaged Product

Install Filter

Filter Lubricant

Prevent Lubricant Leakage

Transmit Shock & Vibration

Remove Filter

Store Disposed Product

Pre-Process Disposed Product

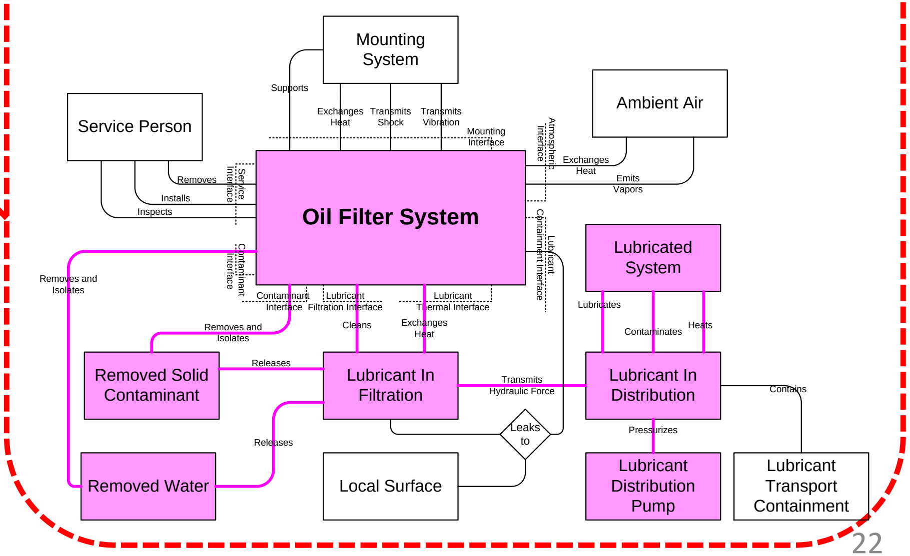
Recycle Disposed Product

Destroy Disposed Product

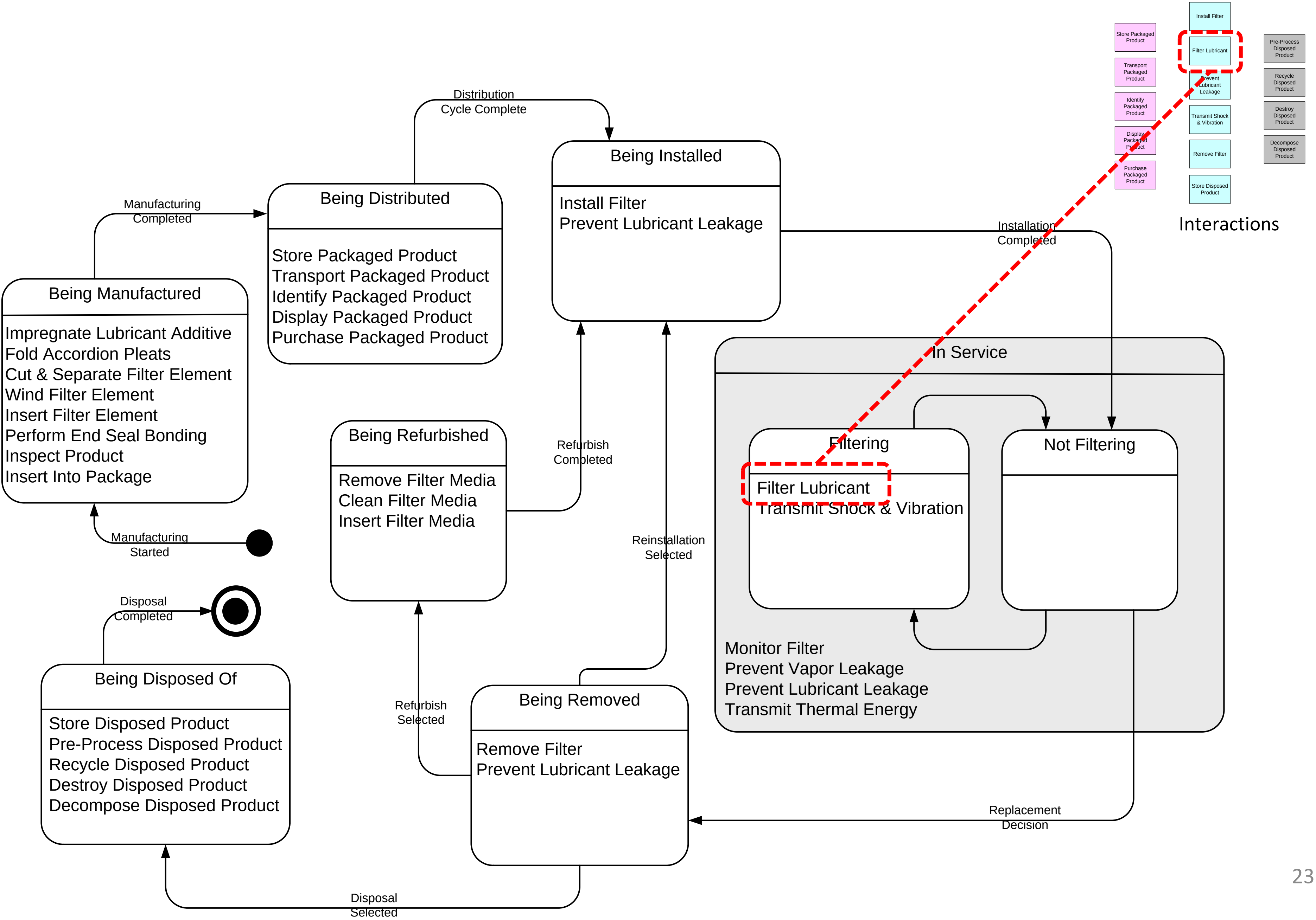
Decompose Disposed Product

- **Trace to external actors/interfaces:**

Interaction: Filter Lubricant



- State Model shows all life cycle states / modes / situations:



Being Manufactured
- Impregnate Lubricant Additive
- Fold Accordion Pleats
- Cut & Separate Filter Element
- Wind Filter Element
- Insert Filter Element
- Perform End Seal Bonding
- Inspect Product
- Insert Into Package

Being Distributed
- Store Packaged Product
- Transport Packaged Product
- Identify Packaged Product
- Display Packaged Product
- Purchase Packaged Product

Being Installed
- Install Filter
- Prevent Lubricant Leakage

In Service
- Monitor Filter
- Prevent Vapor Leakage
- Prevent Lubricant Leakage
- Transmit Thermal Energy

Filtering
- Filter Lubricant
- Transmit Shock & Vibration

Not Filtering

Being Refurbished
- Remove Filter Media
- Clean Filter Media
- Insert Filter Media

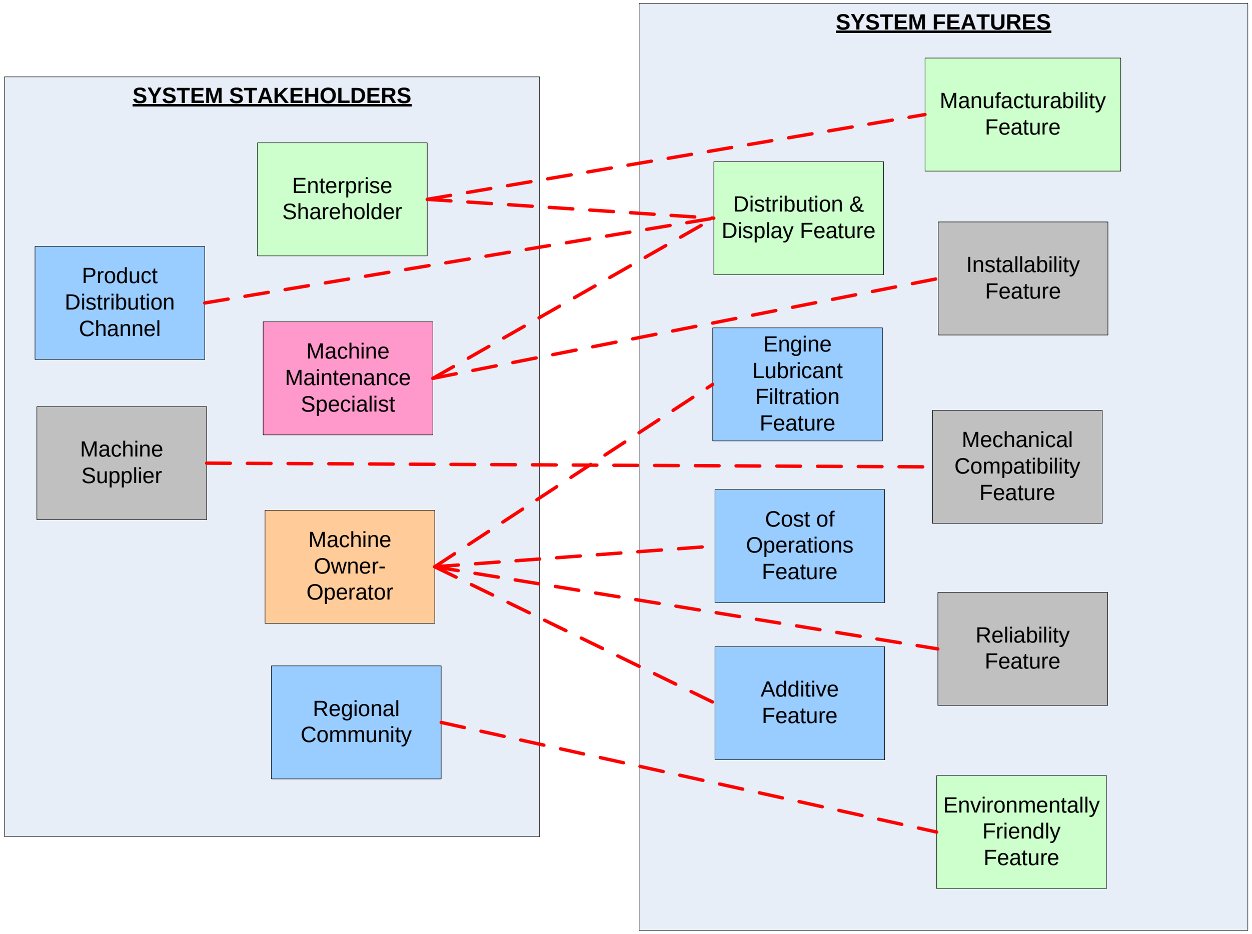
Being Removed
- Remove Filter
- Prevent Lubricant Leakage

Being Disposed Of
- Store Disposed Product
- Pre-Process Disposed Product
- Recycle Disposed Product
- Destroy Disposed Product
- Decompose Disposed Product

Interactions

A Simple Example: Oil Filter

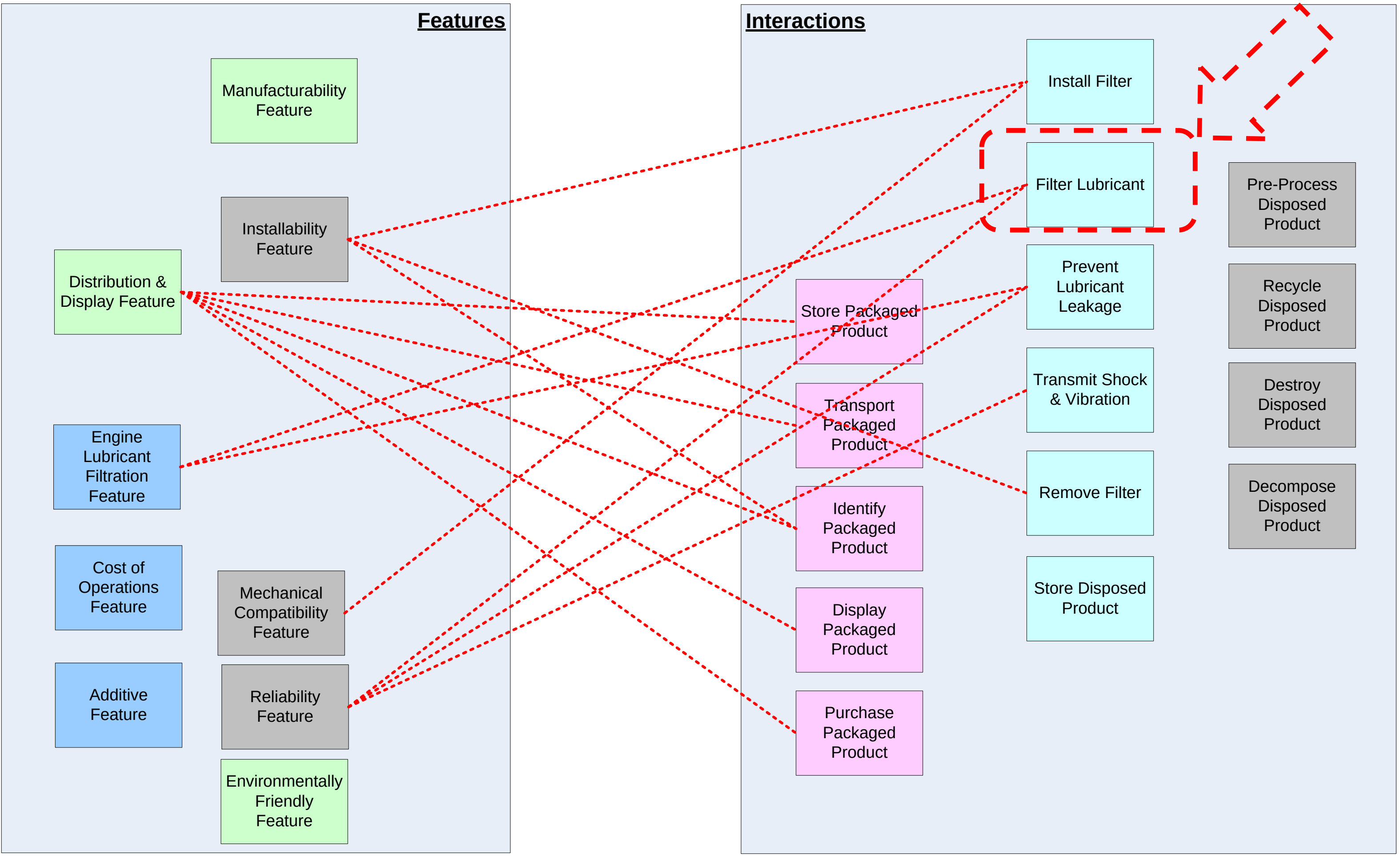
- Stakeholder Feature Model shows all life cycle stakeholders/features
  - Formalizes Stakeholder Requirements in stakeholder language, creating a value / fitness / utility trade space:



Each Feature also has Feature Attributes further quantifying Stakeholder value issues.

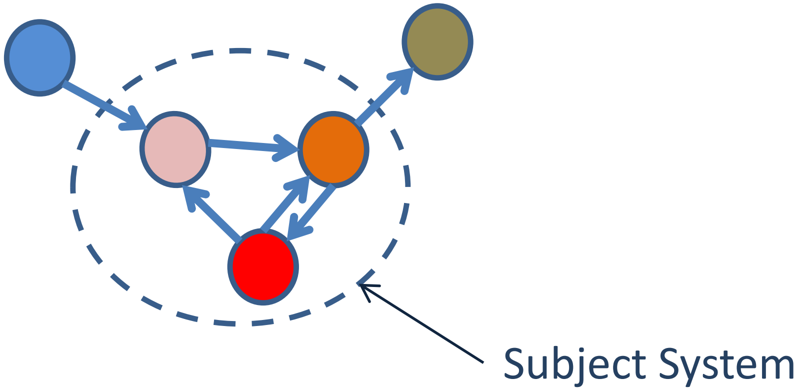
A Simple Example: Oil Filter

- Trace of Interactions to Stakeholder Features:



Are there “other” requirements?

- It is somewhat traditional to divide requirements into categories, such as “functional” and “non-functional”
  - With “non-functional” including reliability, capacity or performance, etc.
- Categorization of requirements is fine, but should not obscure the physical fact that:
  - All system requirements are descriptions of externally- visible behavior, including reliability, capacity, or otherwise
  - All system requirements describe behavior during interactions of the system with external actors—including reliability, capacity, or otherwise

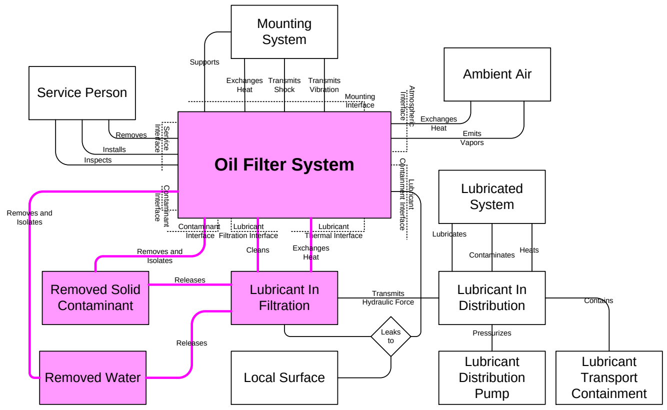


Appearance of Requirement Statements: Examples

OF-50: “For a Return Lubricant stream of [Lubricant Viscosity Range] and [Lubricant Pressure Range], the Oil Filter shall separate Filtered Contaminant particles from the Lubricant output stream, according to the [Filter Particle Size Distribution Profile].”

OF-51: “The Oil Filter shall operate at lubricant pressure of [Max Lubricant Pressure] with structural failure rates less than [Max Structural Failure Rate] over an in-service life of [Min Service Life].”

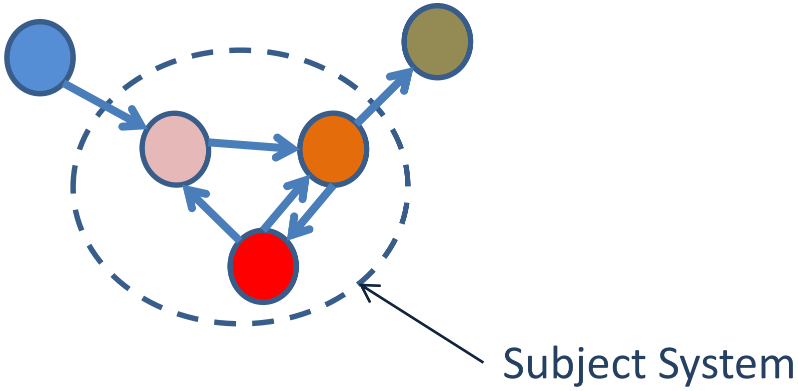
Interaction: Filter Lubricant



Next Illustrative Problem Addressed

2. Unexpected system problems caused by environmental systems, including humans

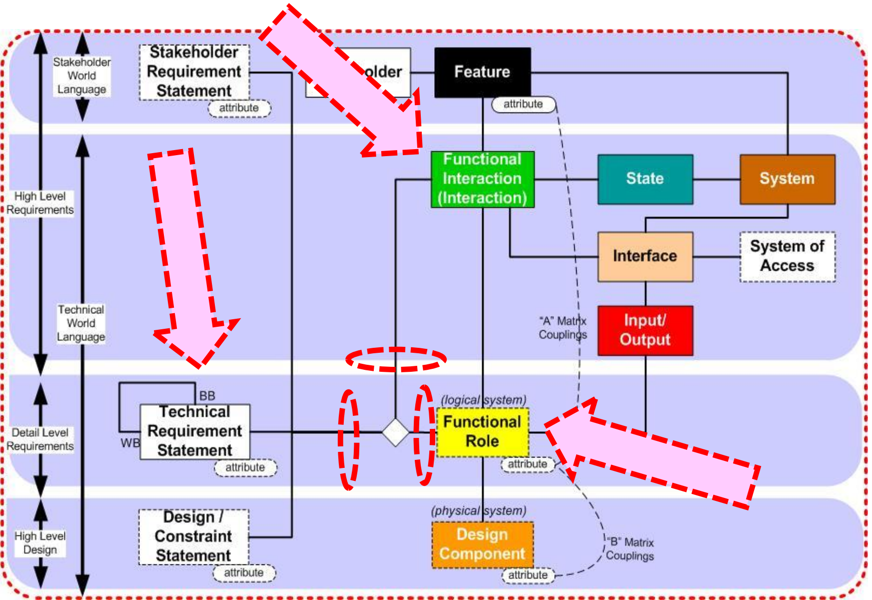
- Interactions-based view :
  - The expected behavior of systems that can impact my system can be more readily and clearly defined, for verification in advance with the responsible parties
  - The expected behavior of human users/ maintainers can be more readily and clearly defined, for early verification as to feasibility, as well as creation of training and documentation

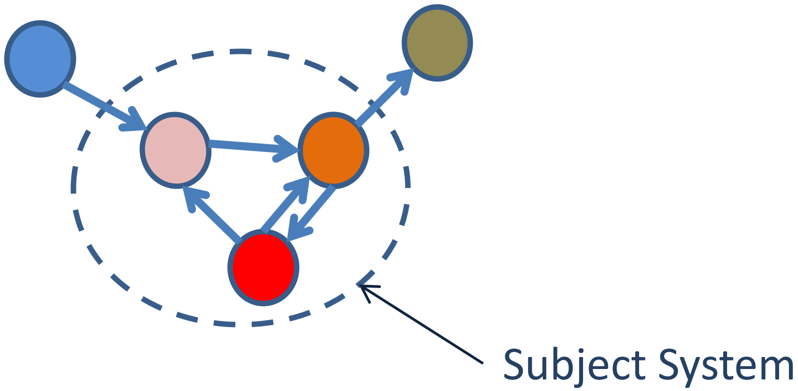


Let’s see how this is done . . . .

Interaction Models Can Coordinate System Compatibilities

- Interactions emphasize the idea that the two (or more) actors participating in the Interaction each have “roles” to play in the overall outcome.
- If one of these actors meets its requirements, but the other does not (even if external), the overall result can be bad—because of the emergent Interaction.
- Whether you call an external actor’s expected behavior “requirements” or “assumptions”, the same point applies.
- The key point is that these component behaviors are associated with a common entity (the Interaction) for coordination purposes.





Example

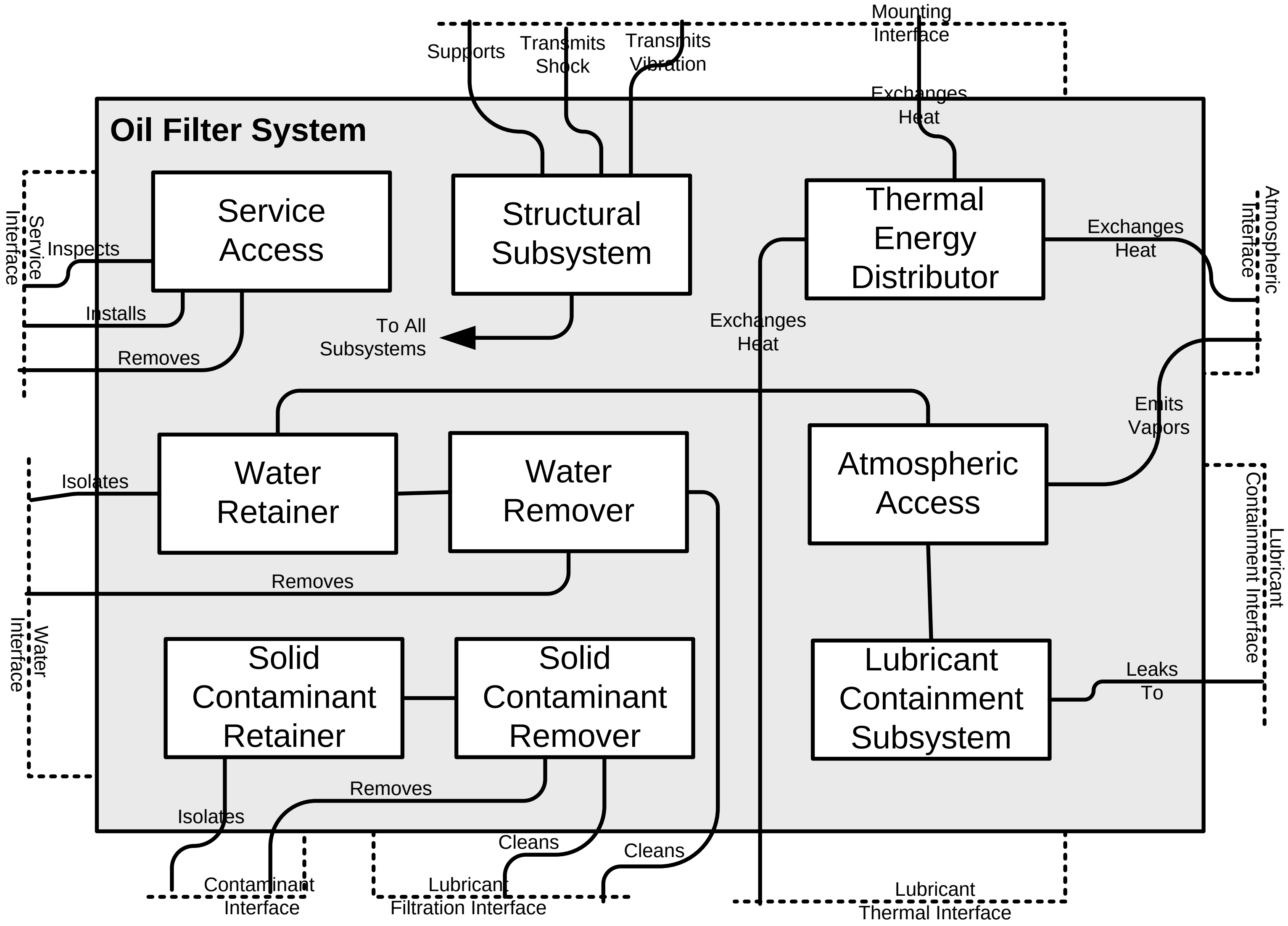
Interaction	Role	Requirement (Required or Assumed Behavior)
Filter Lubricant	Oil Filter System	For a Return Lubricant stream of [Lubricant Viscosity Range] and [Lubricant Pressure Range], the Oil Filter shall separate Filtered Contaminant particles from the Lubricant output stream, according to the [Filter Particle Size Distribution Profile].
Filter Lubricant	Lubricant in Filtration	The Lubricant in Filtration shall have viscosity within the [Lubricant Viscosity Range].
Filter Lubricant	Lubricant Distribution Pump	The Pump shall maintain oil pressure within the [Lubricant Pressure Range].
Install Filter	Oil Filter System	The Oil Filter shall be manually installable in ten minutes or less, using only a screwdriver.
Install Filter	Oil Filter System	The Oil Filter shall have installation instructions printed on its exterior surface, in English
Install Filter	Service Person	The Service Person shall have the visual acuity and hand strength of an average 40 year old adult.
Install Filter	Service Person	The Service Person shall be capable of reading English at the tenth grade level.

3. Understanding causes of emergent behavior

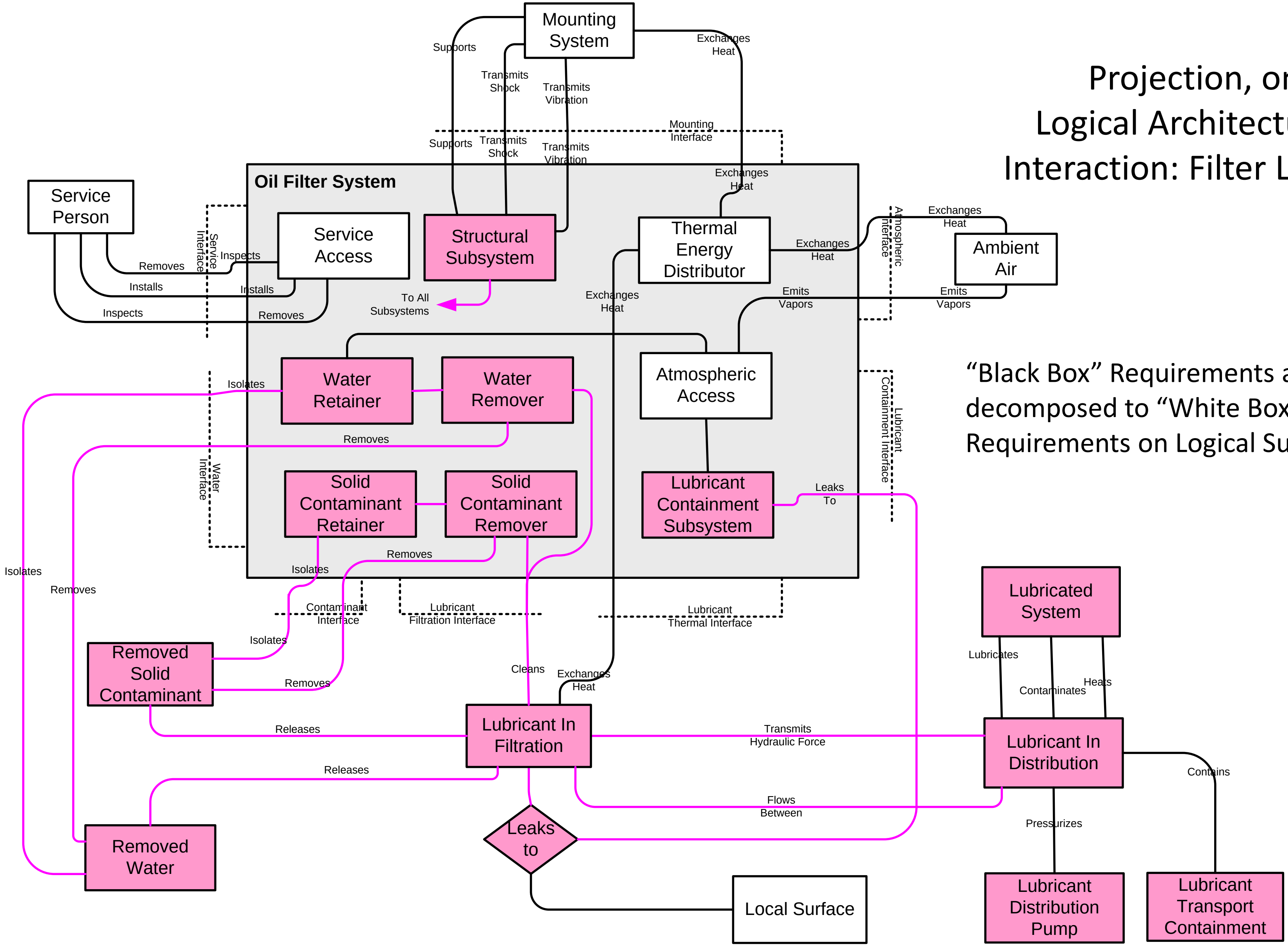
- Interactions-based view:
  - "Whats" are more objectively differentiated (as external physical Interactions), from "hows" (as internal or other aspects).
  - Internal Interactions can be systematically discovered and analyzed, and are the basis for all emergent external behavior
  - Design review is put on a more technical, objective, and transparent / explicit basis by Interaction models
  - Subject matter expertise and experience can be significantly leveraged with explicit Interaction Patterns
  - Proposed changes to designs can be more systematically analyzed in terms of their likely impacts.

- Let's see why the above are so . . .

Logical Architecture Model Provides a Decomposition of External Behavior, Before Allocations to Physical Entities



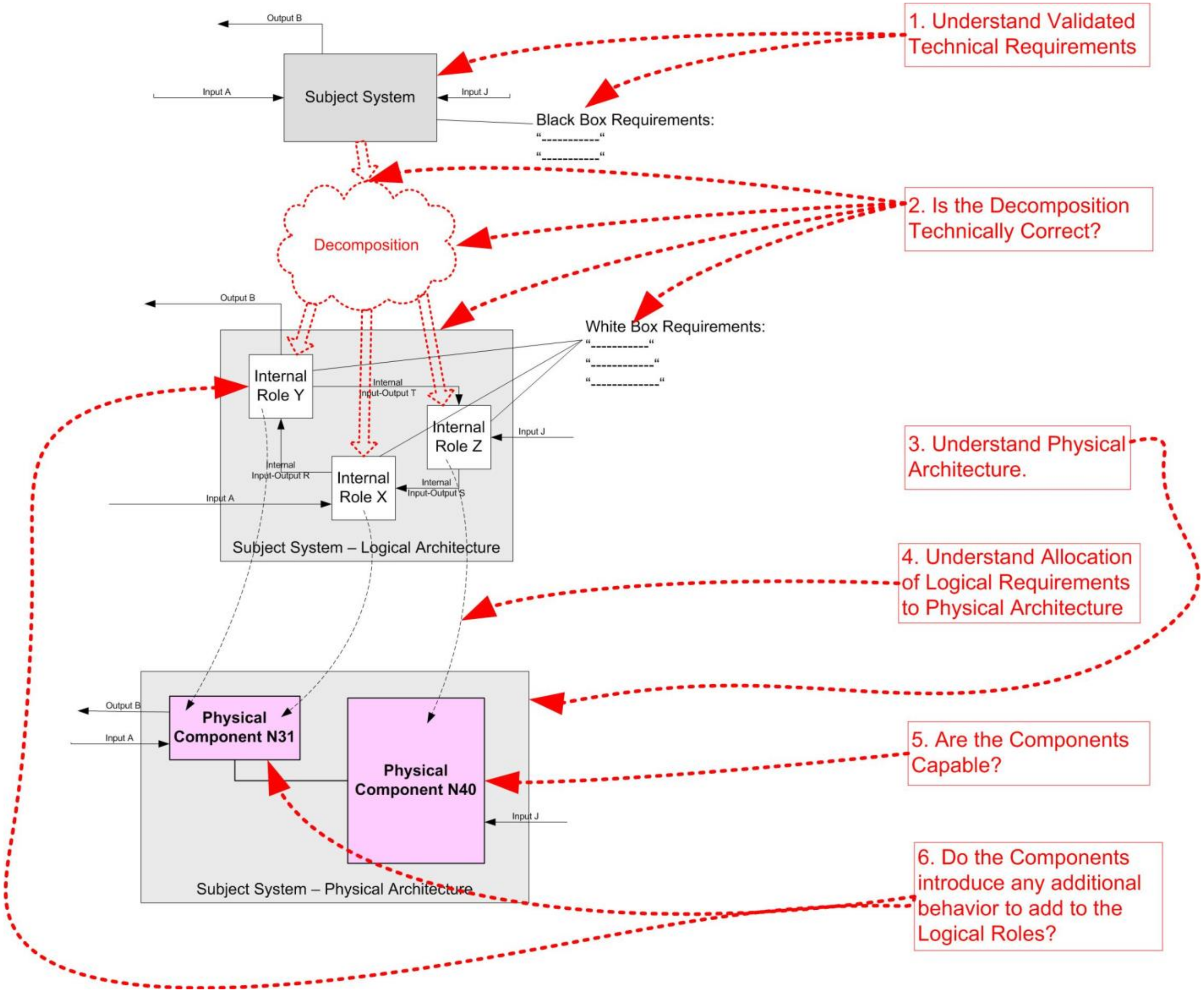
Projection of External Interactions onto Logical Architecture Provides a Framework for Analysis of Internal Interactions



Projection, onto Logical Architecture, of Interaction: Filter Lubricant

"Black Box" Requirements are decomposed to "White Box" Requirements on Logical Subsystems.

Facilitates Improved Design Review: For Each Interaction . . .



1. Understand Validated Technical Requirements

2. Is the Decomposition Technically Correct?

3. Understand Physical Architecture.

4. Understand Allocation of Logical Requirements to Physical Architecture

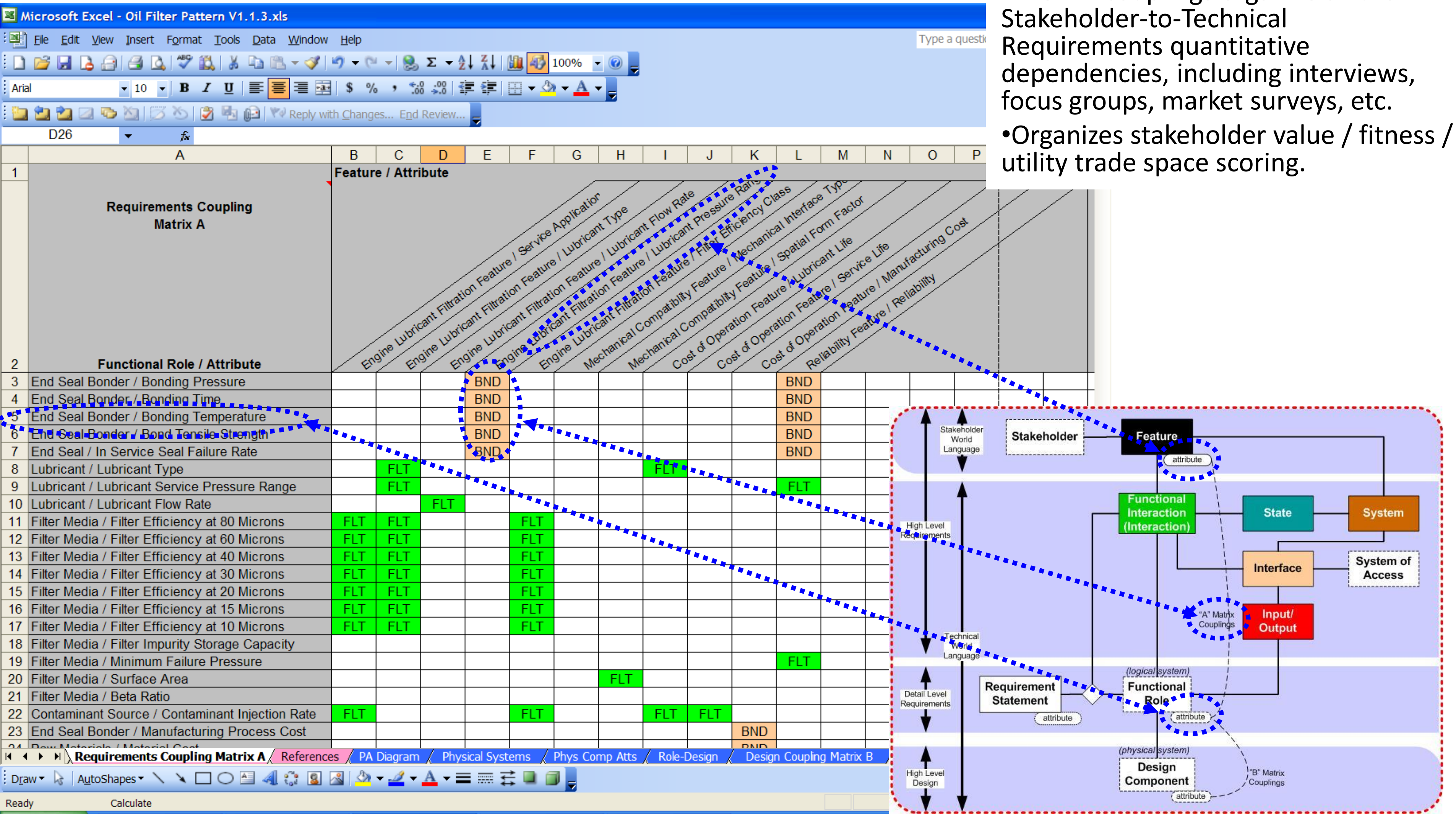
5. Are the Components Capable?

6. Do the Components introduce any additional behavior to add to the Logical Roles?

Note that pure traditional Requirement Statement decomposition is weaker than modeling Interactions between subsystems, which generates the same decompositions but also the interactions that connect them.

Attribute (Parameter) Couplings--Requirements

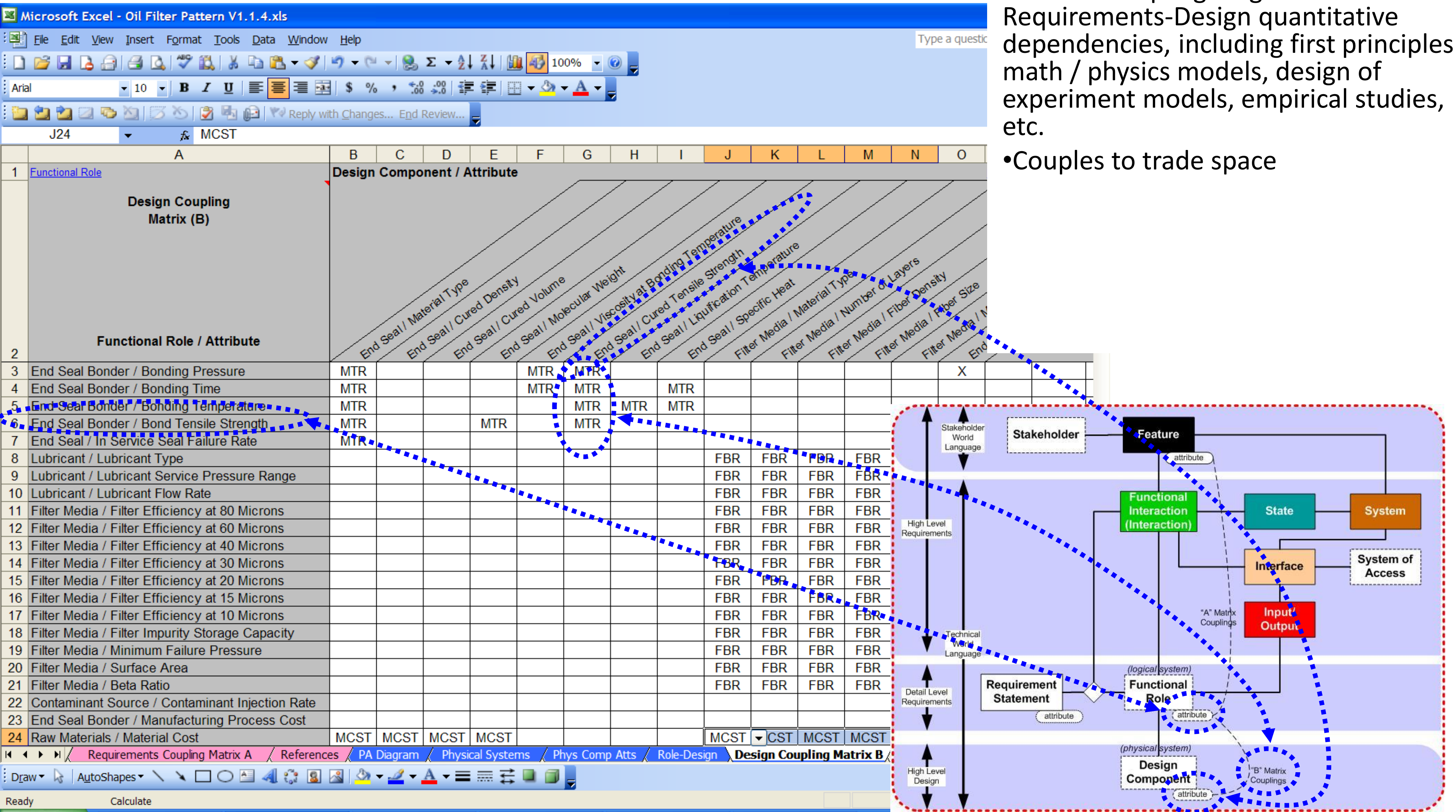
- The "A" couplings organize all the Stakeholder-to-Technical Requirements quantitative dependencies, including interviews, focus groups, market surveys, etc.
- Organizes stakeholder value / fitness / utility trade space scoring.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	**Requirements Coupling Matrix A**	**Feature / Attribute**														
2	**Functional Role / Attribute**	Engine Lubricant Filtration Feature / Service Application	Engine Lubricant Filtration Feature / Lubricant Type	Engine Lubricant Filtration Feature / Lubricant Flow Rate	Engine Lubricant Filtration Feature / Lubricant Pressure Rang	Engine Lubricant Filtration Feature / Filter Efficiency Class	Mechanical Compatiblity Feature / Mechanical Interface Typ	Mechanical Compatiblity Feature / Spatial Form Factor	Cost of Operation Feature / Lubricant Life	Cost of Operation Feature / Service Life	Cost of Operation Feature / Manufacturing Cost	Reliability Feature / Reliability				
3	End Seal Bonder / Bonding Pressure				BND							BND				
4	End Seal Bonder / Bonding Time				BND							BND				
5	End Seal Bonder / Bonding Temperature				BND							BND				
6	End Seal Bonder / Bond Tensile Strength				BND							BND				
7	End Seal / In Service Seal Failure Rate				BND							BND				
8	Lubricant / Lubricant Type		FLT						FLT							
9	Lubricant / Lubricant Service Pressure Range		FLT									FLT				
10	Lubricant / Lubricant Flow Rate			FLT												
11	Filter Media / Filter Efficiency at 80 Microns	FLT	FLT			FLT										
12	Filter Media / Filter Efficiency at 60 Microns	FLT	FLT			FLT										
13	Filter Media / Filter Efficiency at 40 Microns	FLT	FLT			FLT										
14	Filter Media / Filter Efficiency at 30 Microns	FLT	FLT			FLT										
15	Filter Media / Filter Efficiency at 20 Microns	FLT	FLT			FLT										
16	Filter Media / Filter Efficiency at 15 Microns	FLT	FLT			FLT										
17	Filter Media / Filter Efficiency at 10 Microns	FLT	FLT			FLT										
18	Filter Media / Filter Impurity Storage Capacity															
19	Filter Media / Minimum Failure Pressure											FLT				
20	Filter Media / Surface Area							FLT								
21	Filter Media / Beta Ratio															
22	Contaminant Source / Contaminant Injection Rate	FLT				FLT			FLT	FLT						
23	End Seal Bonder / Manufacturing Process Cost										BND					
24	Raw Materials / Material Cost										BND					

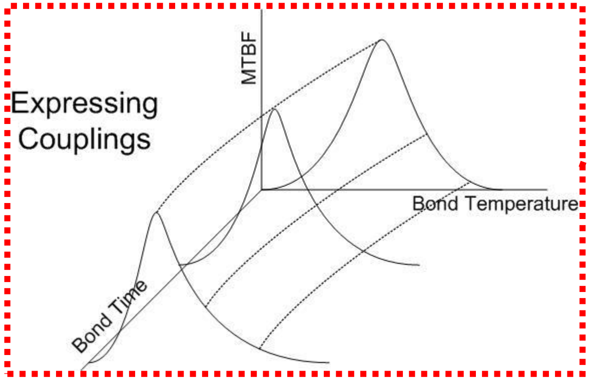
Attribute (Parameter) Couplings--Design

- The "B" couplings organize all the Requirements-Design quantitative dependencies, including first principles math / physics models, design of experiment models, empirical studies, etc.
- Couples to trade space



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Functional Role	Design Component / Attribute													
2	Design Coupling Matrix (B) Functional Role / Attribute	End Seal / Material Type	End Seal / Cured Density	End Seal / Cured Volume	End Seal / Molecular Weight	End Seal / Viscosity at Bonding Temperature	End Seal / Cured Tensile Strength	End Seal / Liquification Temperature	End Seal / Specific Heat	Filter Media / Material Type	Filter Media / Number of Layers	Filter Media / Fiber Density	Filter Media / Fiber Size	Filter Media / M	End
3	End Seal Bonder / Bonding Pressure	MTR				MTR	MTR								X
4	End Seal Bonder / Bonding Time	MTR				MTR	MTR		MTR						
5	End Seal Bonder / Bonding Temperature	MTR					MTR	MTR	MTR						
6	End Seal Bonder / Bond Tensile Strength	MTR			MTR		MTR								
7	End Seal / In Service Seal Failure Rate	MTR													
8	Lubricant / Lubricant Type									FBR	FBR	FBR	FBR		
9	Lubricant / Lubricant Service Pressure Range									FBR	FBR	FBR	FBR		
10	Lubricant / Lubricant Flow Rate									FBR	FBR	FBR	FBR		
11	Filter Media / Filter Efficiency at 80 Microns									FBR	FBR	FBR	FBR		
12	Filter Media / Filter Efficiency at 60 Microns									FBR	FBR	FBR	FBR		
13	Filter Media / Filter Efficiency at 40 Microns									FBR	FBR	FBR	FBR		
14	Filter Media / Filter Efficiency at 30 Microns									FBR	FBR	FBR	FBR		
15	Filter Media / Filter Efficiency at 20 Microns									FBR	FBR	FBR	FBR		
16	Filter Media / Filter Efficiency at 15 Microns									FBR	FBR	FBR	FBR		
17	Filter Media / Filter Efficiency at 10 Microns									FBR	FBR	FBR	FBR		
18	Filter Media / Filter Impurity Storage Capacity									FBR	FBR	FBR	FBR		
19	Filter Media / Minimum Failure Pressure									FBR	FBR	FBR	FBR		
20	Filter Media / Surface Area									FBR	FBR	FBR	FBR		
21	Filter Media / Beta Ratio									FBR	FBR	FBR	FBR		
22	Contaminant Source / Contaminant Injection Rate														
23	End Seal Bonder / Manufacturing Process Cost														
24	Raw Materials / Material Cost	MCST	MCST	MCST	MCST					MCST	CST	MCST	MCST		

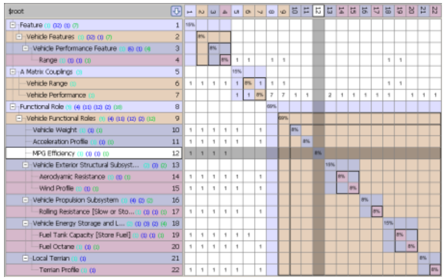
Parameter Coupling Views: DSM and Coupling Representation



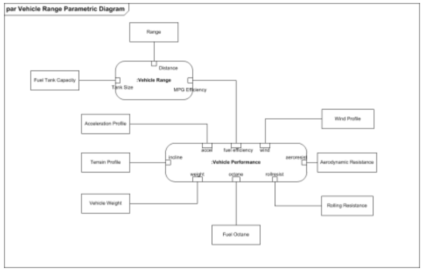
The Coupling Model is a unifying framework integrating all forms of coupling:

- First principles equations
- Empirical datasets
- Graphical relations
- Data tables
- Prose statements
- Fuzzy relationships
- Other

Note that informal use of the term "interactions" for such couplings can be confusing. These couplings are dependencies between parameters, and are not physical interactions between entities.


$root
Feature 1
Vehicle Features 2
Vehicle Performance Feature 3
Range 4
A Matrix Couplings 5
Vehicle Range 6
Vehicle Performance 7
Functional Role 8
Vehicle Functional Roles 9
Vehicle Weight 10
Acceleration Profile 11
MPG Efficiency 12
Vehicle Exterior Structural Subsyst... 13
Aerodyamic Resistance 14
Wind Profile 15
Vehicle Propulsion Subsystem 16
Rolling Resistance [Slow or Sto... 17
Vehicle Energy Storage and L... 18
Fuel Tank Capacity [Store Fuel] 19
Fuel Octane 20
Local Terrian 21
Terrian Profile 22

N2 Coupling Matrix Views (e.g., DSM)



SysML Parametric Coupling Diagram

Implications: What you can do

- Identify the external interactions for your system
  - Check their completeness by tracing each interaction to external Actors/Interfaces, States, and Stakeholder Features
  - Adjust each of the three lists until they include the same Interactions
- Create Requirements Statements for each Interaction:
  - Each should describe the expected input-output relationships during that Interaction
  - You can also list the Requirements (Assumptions) of the external Actors during the same Interactions
- Applies to both new and existing (reverse engineered) systems:
  - There is a high likelihood of finding opportunities to improve understanding and communication.
- Let us know how you do, and good luck!

References

1. W. Schindel, “Requirement Statements Are Transfer Functions: An Insight From Model-based Systems Engineering”, *Proceedings of INCOSE 2005 International Symposium*, (2005).
2. W. Schindel, “What Is the Smallest Model of a System?”, *Proc. of the INCOSE 2011 International Symposium*, International Council on Systems Engineering (2011).
3. W. Schindel, “Systems of Innovation II: The Emergence of Purpose”, *Proceedings of INCOSE 2013 International Symposium* (2013).
4. W. Schindel, S. Peffers, J. Hanson, J. Ahmed, W. Kline, “All Innovation is Innovation of Systems : An Integrated 3-D Model of Innovation Competencies ”, Proc. of ASEE 2011 Conference (2011).
5. W. Schindel, “Pattern-Based Systems Engineering: An Extension of Model-Based SE”, INCOSE IS2005 Tutorial TIES 4, (2005).
6. J. Bradley, M. Hughes, and W. Schindel, “Optimizing Delivery of Global Pharmaceutical Packaging Solutions, Using Systems Engineering Patterns” Proceedings of the INCOSE 2010 International Symposium (2010).
7. W. Schindel, and V. Smith, “Results of applying a families-of-systems approach to systems engineering of product line families”, SAE International, Technical Report 2002-01-3086 (2002).
8. W. Schindel, “The Impact of ‘Dark Patterns’ On Uncertainty: Enhancing Adaptability In The Systems World”, INCOSE Great Lakes 2011 Conference, Dearborn, MI, 2011.
9. W. Schindel and T. Peterson, “Introduction to Pattern-Based Systems Engineering”, *Proceedings of the INCOSE 2013 International Symposium,* June, 2013.
10. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, Version 3.2, International Council on Systems Engineering (2010).

ICTT System Sciences

Speaker

- Bill Schindel (schindel@ictt.com) is president of ICTT System Sciences (www.ictt.com), a systems engineering company. His 40-year engineering career began in mil/aero systems with IBM Federal Systems, Owego, NY, included service as a faculty member of Rose-Hulman Institute of Technology, and founding of three commercial systems-based enterprises. He has led and consulted on improvement of engineering processes within automotive, medical/health care, manufacturing, telecommunications, aerospace, and consumer products businesses. Schindel earned the BS and MS in Mathematics. At the 2005 INCOSE International Symposium, he was recognized as the author of the outstanding paper on Modeling and Tools, and currently co-leads a research project on the science of Systems of Innovation within the INCOSE System Science Working Group. Bill is an INCOSE CSEP, president of the Crossroads of America INCOSE chapter, and general chair of the INCOSE Great Lakes Regional Conference for 2013.