Managing Complex Technical Projects

Day 1: Design Structure Matrix and Process DSM Models

Prof. Steven D. Eppinger

Massachusetts Institute of Technology

Sloan School of Management

eppinger@mit.edu

web.mit.edu/eppinger

www.dsmweb.org

Steven Eppinger

General Motors Leaders for Global Operations Professor
Professor of Management Science and Innovation
Professor of Engineering Systems
Massachusetts Institute of Technology

- **Educational Background:**
  - SB, SM, and ScD in Mechanical Engineering, MIT
- **Teaching:**
  - Product design and development, engineering project management
- **Research Foci:**
  - Product development processes and organizations, complex system development, technical project management, sustainable design
- **Industry Experience:**
  - Aerospace, automotive, consumer electronics, software, telecommunications, medical devices, capital equipment

MIT MANAGEMENT
SLOAN SCHOOL

Program Outline

Day 1: Design Structure Matrix and Process DSM Models

Complex projects, DSM types Process modeling with DSM	*break*	Sequencing methods Planned vs unplanned iterations	*break*	Process DSM building Application discussions

Day 2: System Architecture and Organization DSM Models

Organization architecture DSM System architecture DSM	*break*	DSM building and clustering Combined system + org DSM	*break*	System readiness DSM Application discussions

Day 3: Agile Development Methods

Background and manifesto Agile techniques you can use	*break*	Team-based agile (Scrum) Scaled agile methods	*break*	Agile coaching session Course wrap-up

Brief Personal Introductions

1. Your name
2. Company you work for
3. Current role

Optional:

4. Key question you would like to answer this week

Scale and Scope of Engineering Projects

10 people

5 suppliers

2 years

3 sites

AvaTech Snow Profiler

1000 people

500 suppliers

10 years

25 sites

Virgin Galactic Spaceship

Industrial Examples (and Research Sponsors)

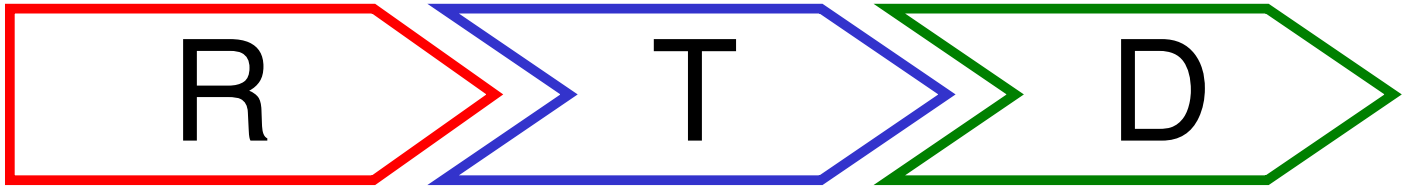
Discussion Question

What drives the ***complexity*** of large, technical projects?

5 minutes breakout room discussion

Use the Google Slides link which is in the Chat

Research and Development



Basic Research

- Discovery process
- No set timing
- Unpredictable returns
- Long term

DSM and Agile methods apply somewhat to basic research projects.

Technology Development

- Loosely structured
- Difficult to plan
- Less predictable
- Medium term

DSM and Agile methods apply somewhat to technology development projects.

Product Development

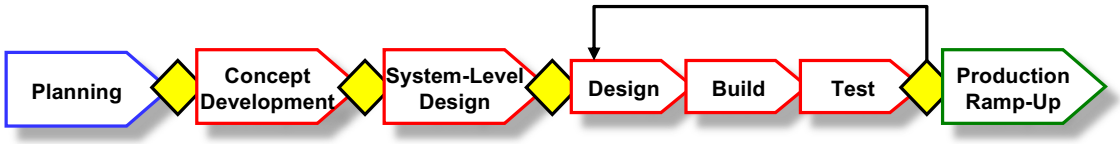
- Structured methods
- Planned timing
- Predictable outcome
- Short term

DSM and Agile methods apply very well to product development projects.

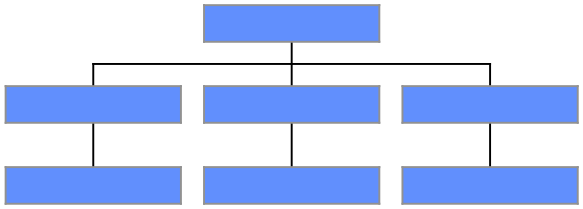
Three Perspectives in Managing Development of Complex Products

- **Product/System**

- **Process**

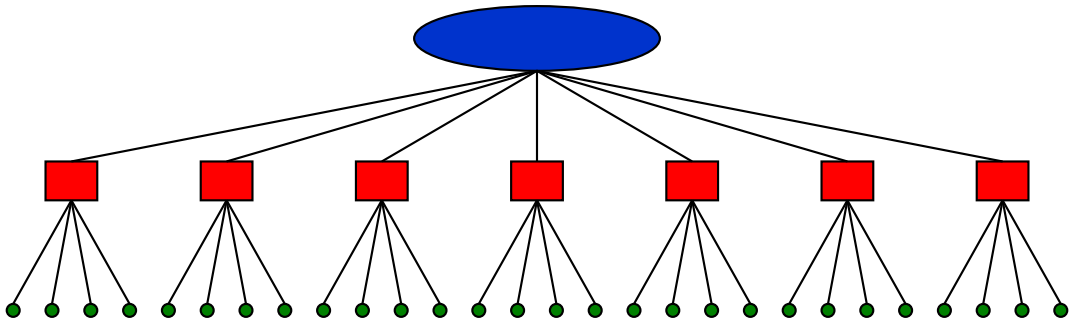


- **Organization**



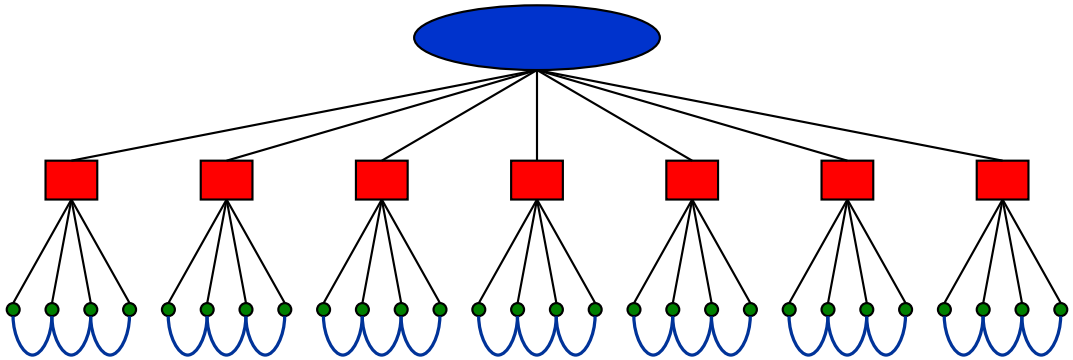
Decomposition:
The Key to Managing Complexity

- Decompose a complex product/system into sub-systems and components
- Decompose a complex process into sub-processes and tasks
- Decompose a large organization into teams and individuals



Decompositions Exhibit Structure (Architecture)

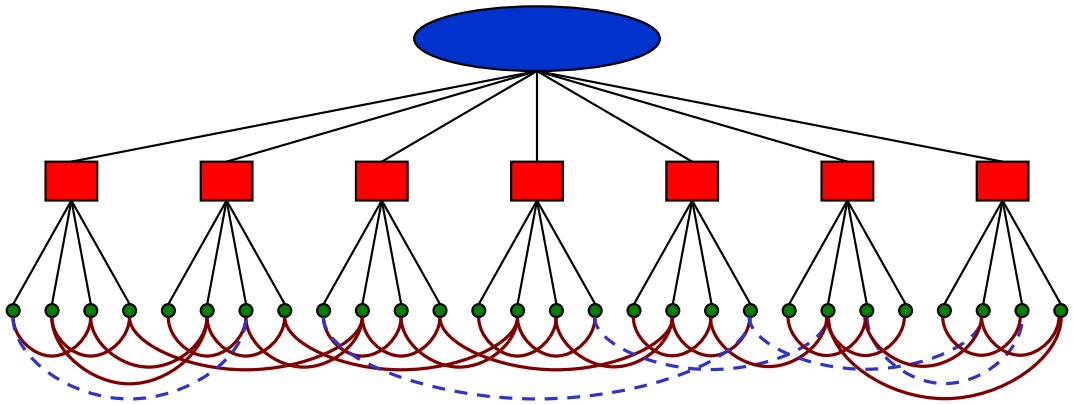
- The pattern of interactions between the decomposed elements define the architecture
  - Product/system architecture
  - Process architecture
  - Organization architecture



"simple architecture"

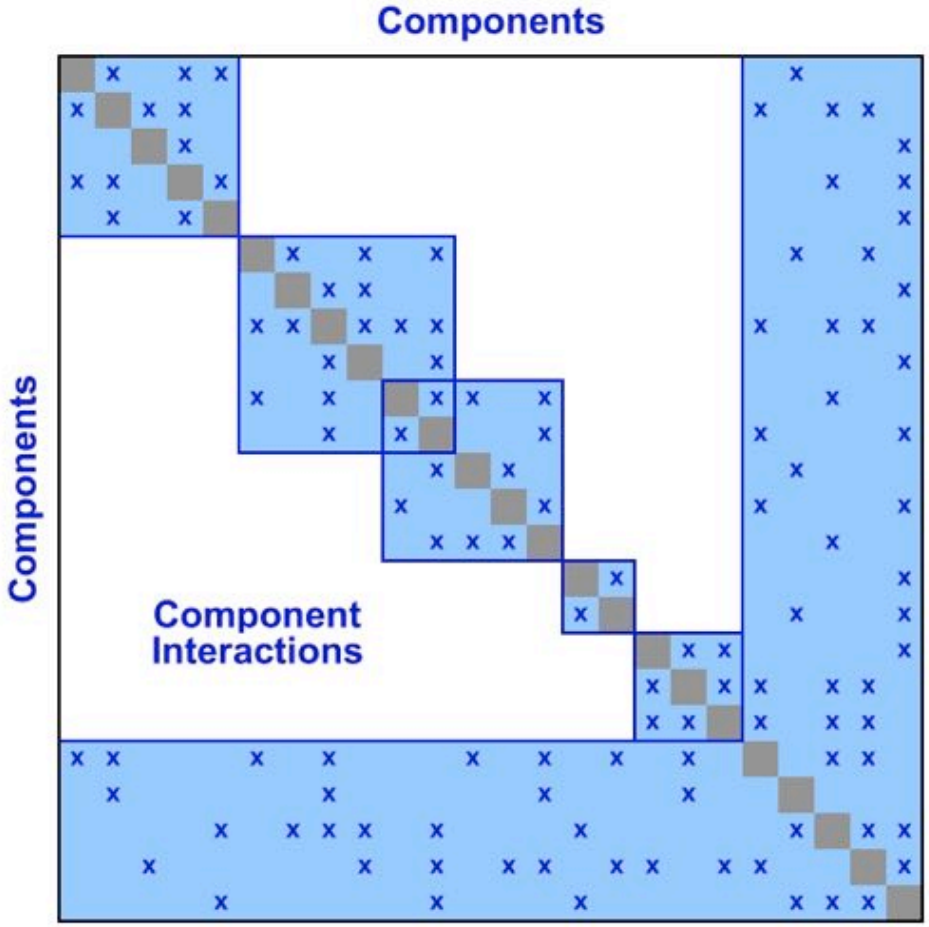
Decompositions Exhibit Structure (Architecture)

- The pattern of interactions between the decomposed elements define the architecture
  - Product/system architecture
  - Process architecture
  - Organization architecture

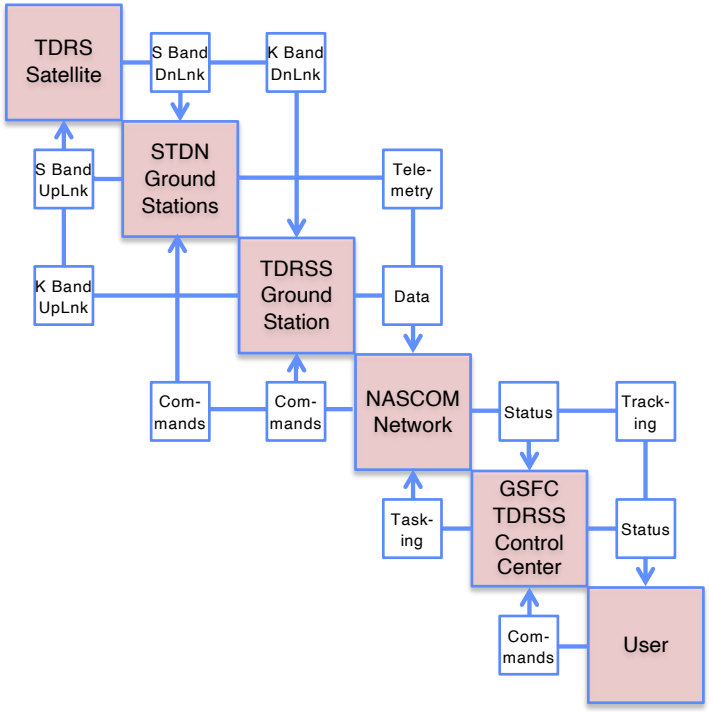


"complex architecture"

Design Structure Matrix (DSM): Product/System Architecture

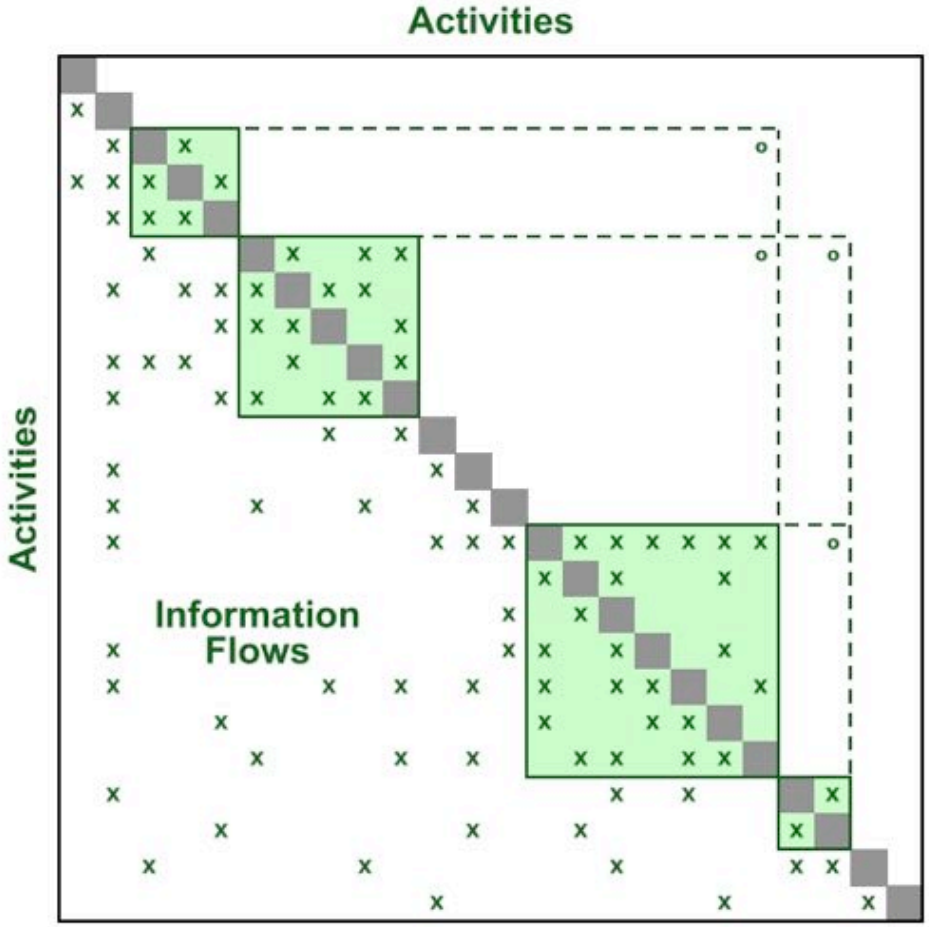


Systems Engineering N^2 Chart

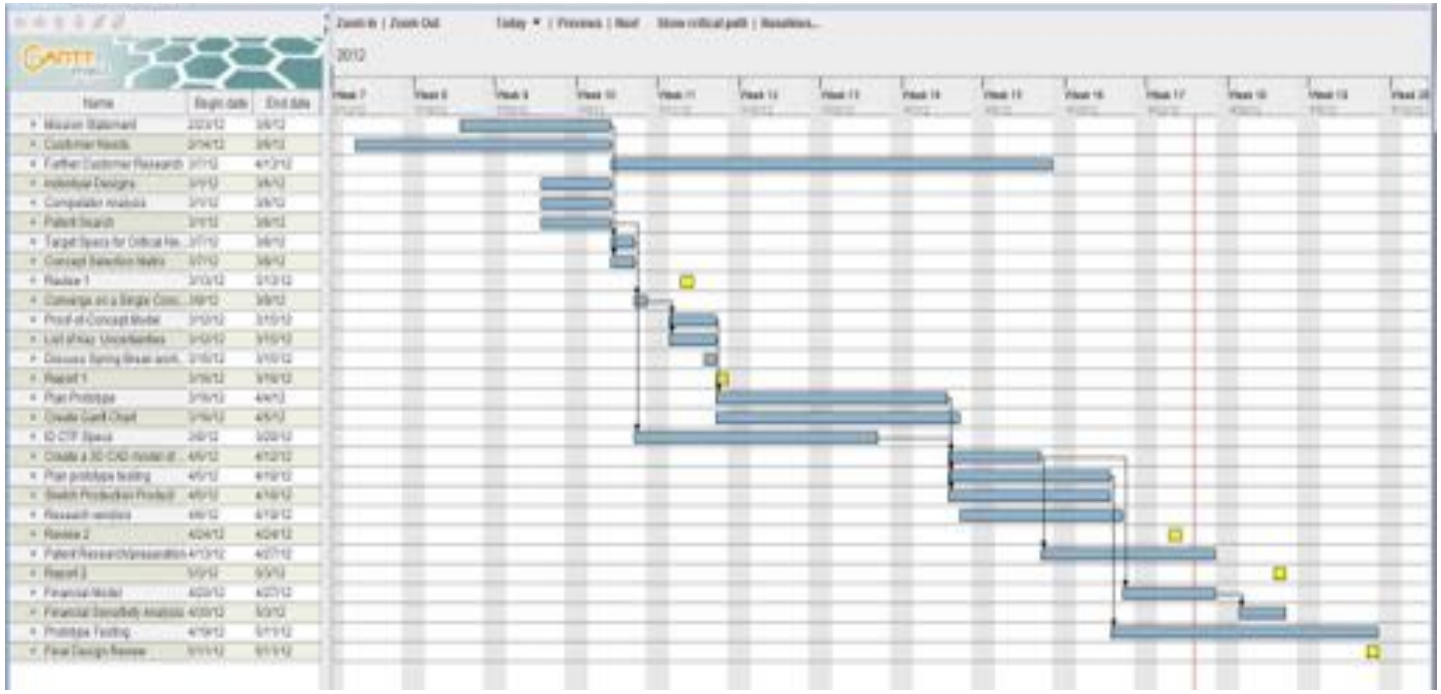


Ref: R.J. Lano, *A Technique for Software and Systems Design,* 1979

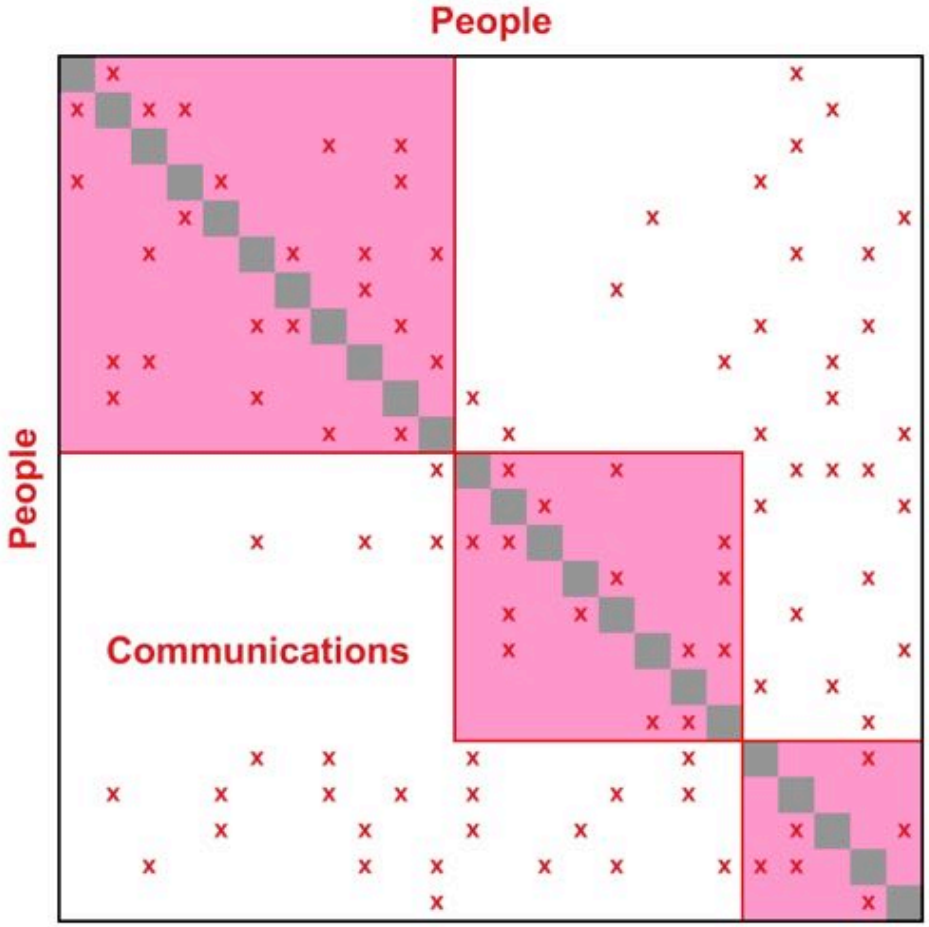
Design Structure Matrix (DSM): Process Architecture



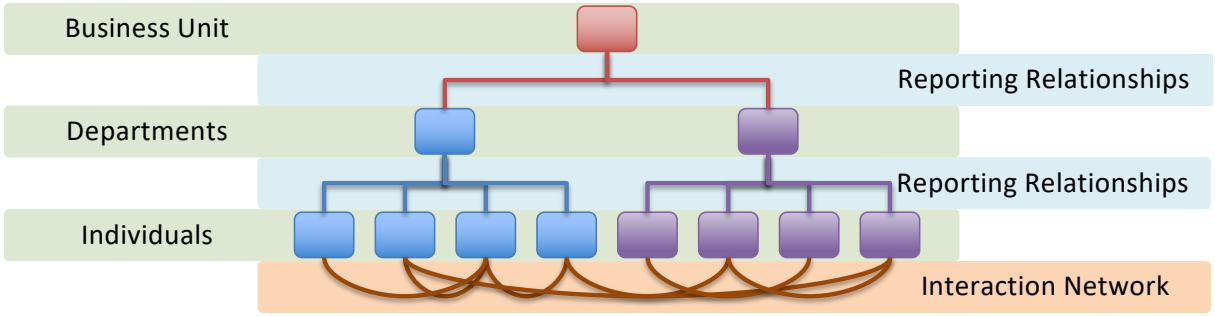
Gantt Chart



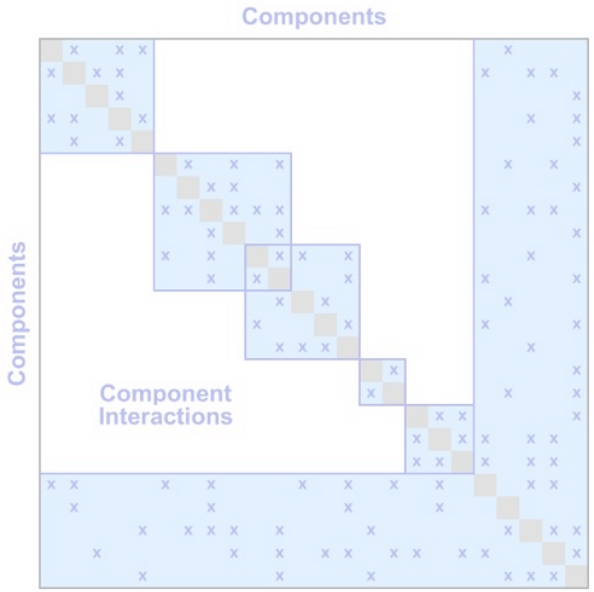
Design Structure Matrix (DSM): Organization Architecture



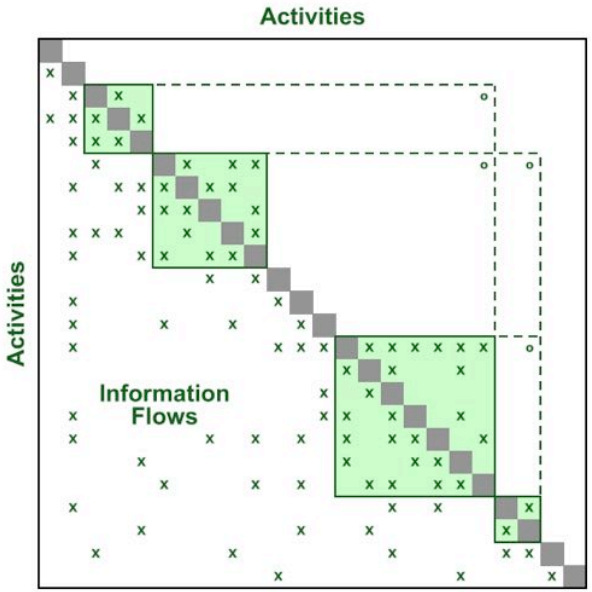
Organization Chart



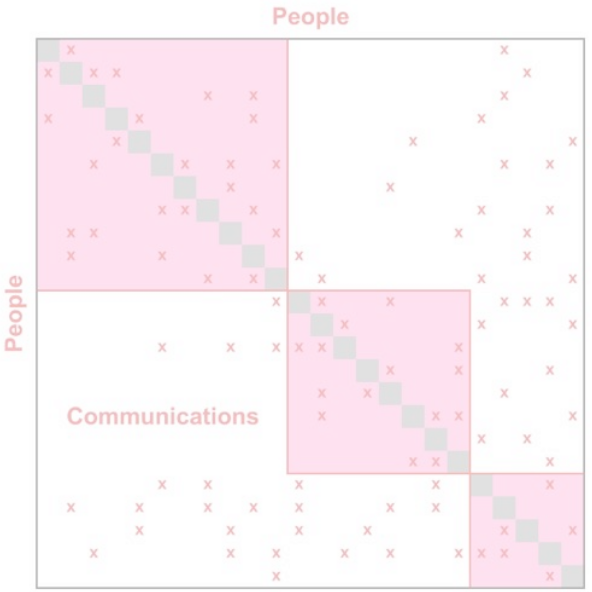
Three Primary DSM Types



Product Architecture
DSM



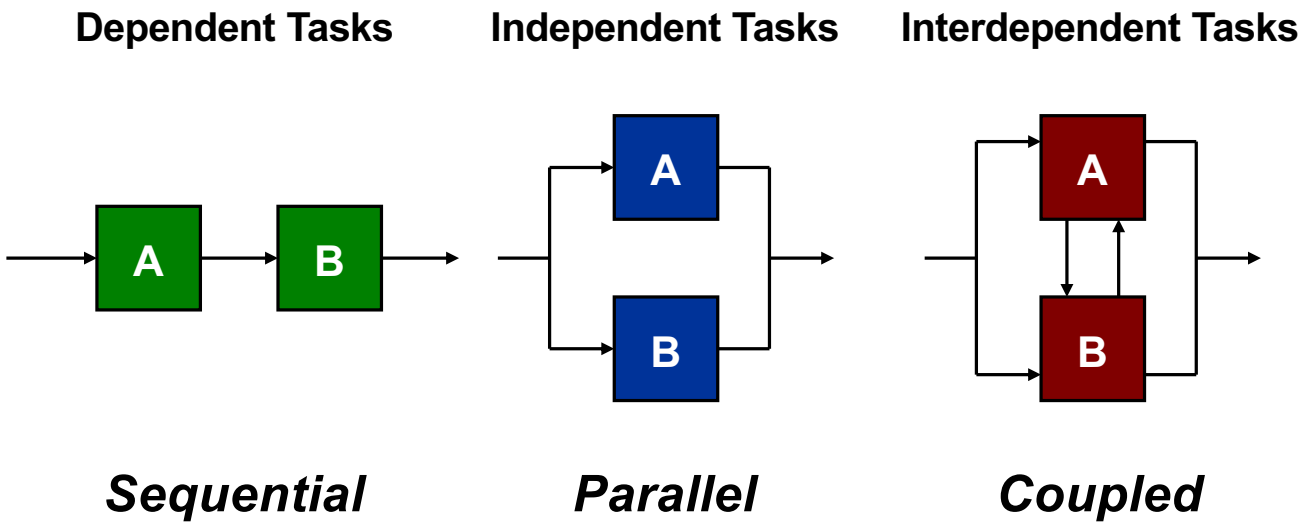
Process Architecture
DSM



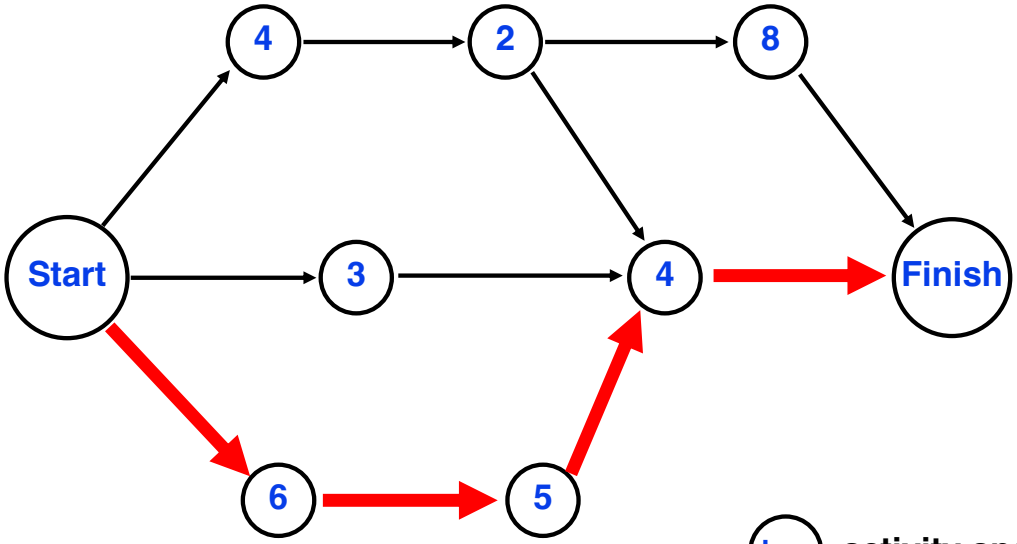
Organization Architecture
DSM

Sequencing Tasks in Projects

Three Possible Sequences of Two Tasks



PERT and CPM Charts



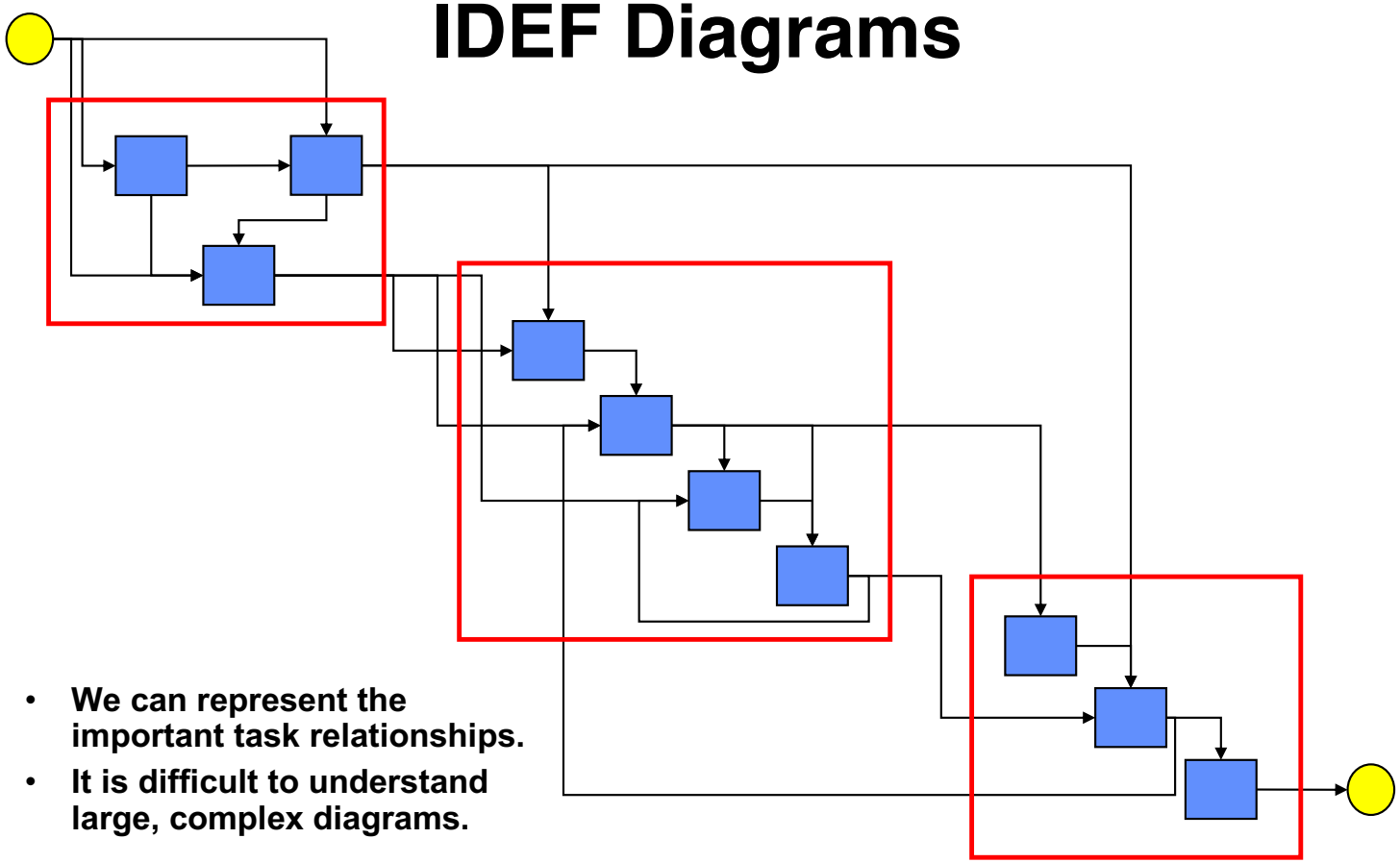
- Simple network diagrams are easy to understand.
- We cannot represent the coupled/iterative task relationships.

days activity and duration

activity precedence

critical path

IDEF Diagrams



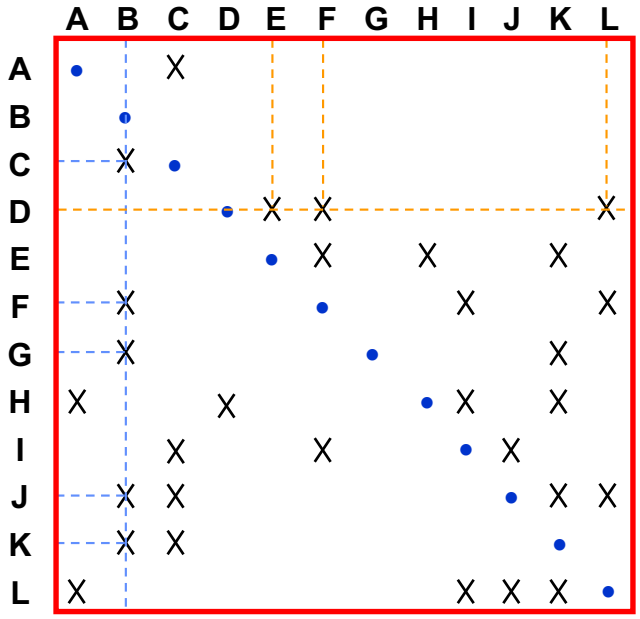
- We can represent the important task relationships.
- It is difficult to understand large, complex diagrams.

The Design Structure Matrix:
An Information Exchange Model

IR Convention:

- Inputs in rows
- Outputs in columns

OUTPUTS
INPUTS



Interpretation:

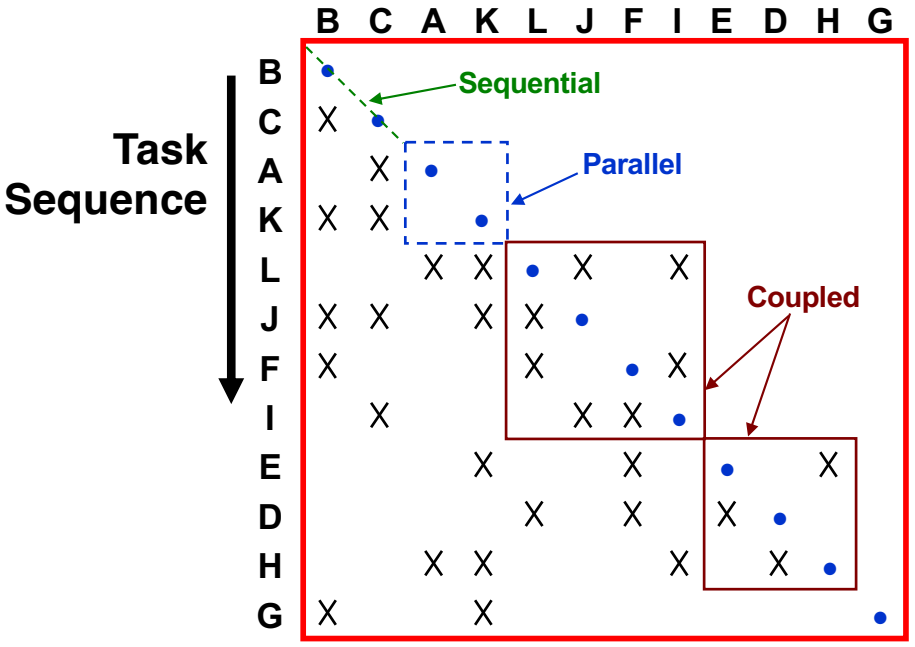
- Task D needs information from tasks E, F, and L.
- Task B feeds information to tasks C, F, G, J, and K.

FEEDS
NEEDS

Earliest DSM Paper:
Donald V. Steward, Aug. 1981
IEEE Trans. on Eng'g Mgmt.

The Design Structure Matrix

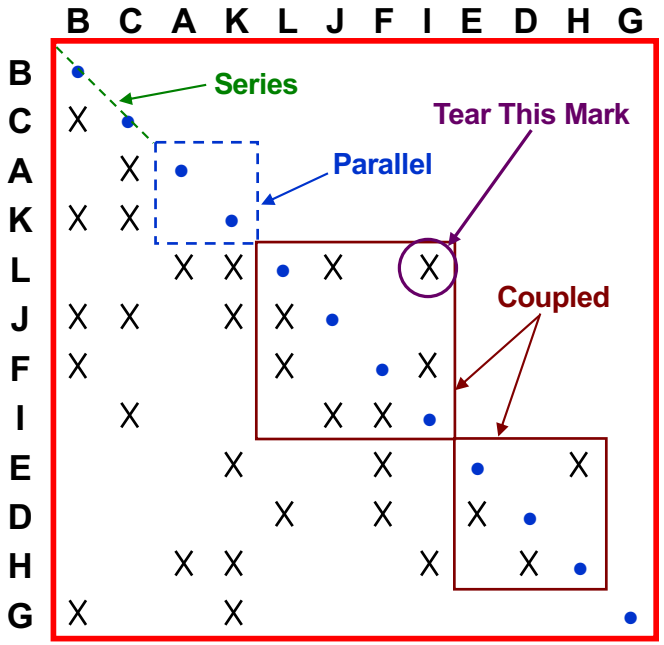
(Sequenced)



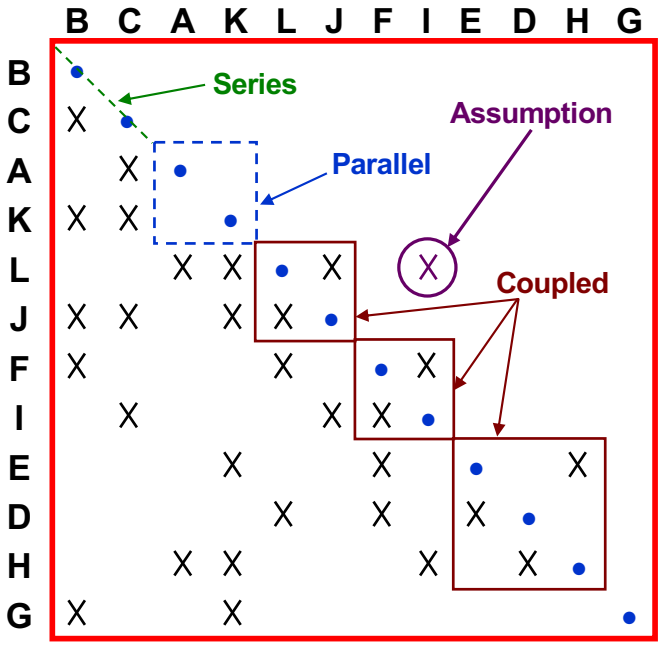
Notes:

- Coupled tasks can be identified uniquely.
- Display of the matrix can be formatted to emphasize certain features of the process flow.
- DSM analysis software implements sequencing algorithms.

Tearing Marks in the DSM



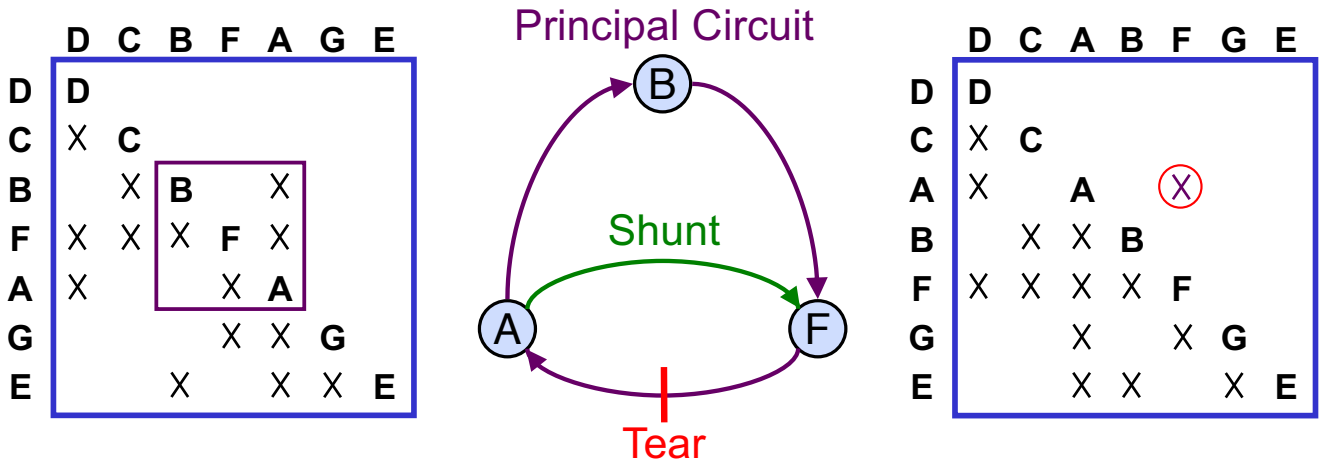
Tear the marks which break the coupled block into smaller ones or make it sequential.



Torn marks may become

- Assumptions
- Feedbacks
- Controls for the process

Finding the Principal Circuit and Shunts Helps Decide Where to Tear



- The suggested tear, between F and A, breaks both of the circuits.
- DSM analysis software can identify principal circuits and suggest tears.

DSM Sequencing

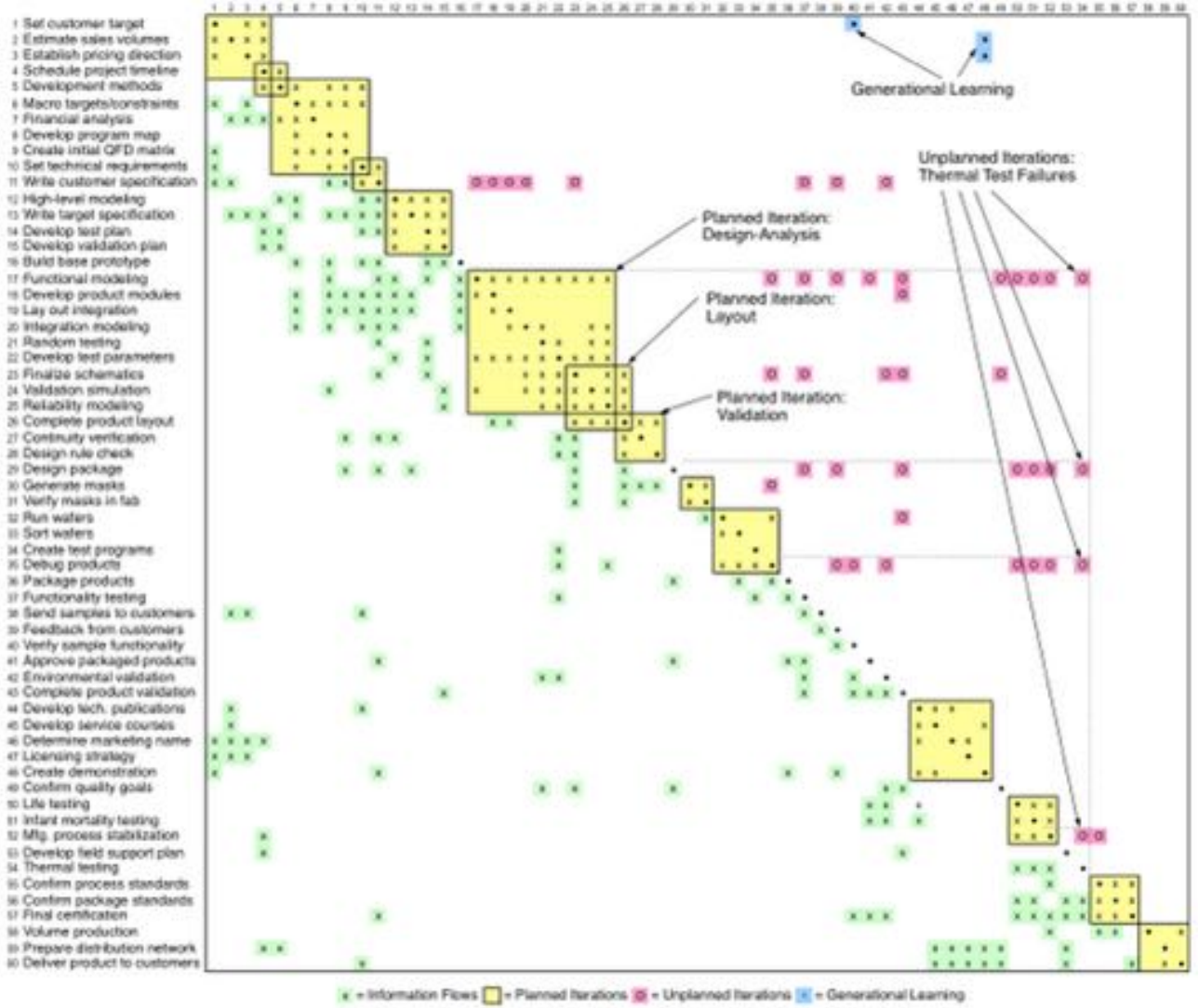
Algorithmic Approach

1. Find any empty rows. Move these up to the front.
2. Find any empty columns. Move these to the end.
3. Find any loop, collapse it, and schedule it as above if possible.
4. Repeat steps 1-3 until all tasks and loops are sequenced.

Sequencing Exercise

	A	B	C	D	E	F	G	H	I	J	
A	A		X		X					X	A
B	X	B	X					X	X		B
C			C								C
D				D	X		X			X	D
E	X				E					X	E
F	X	X	X	X		F	X				F
G				X			G			X	G
H		X			X			H		X	H
I	X	X	X						I		I
J			X		X					J	J
	A	B	C	D	E	F	G	H	I	J	

Process DSM: Semiconductor Development



Cross-Functional Interactions: Real Estate Development

Responsibility

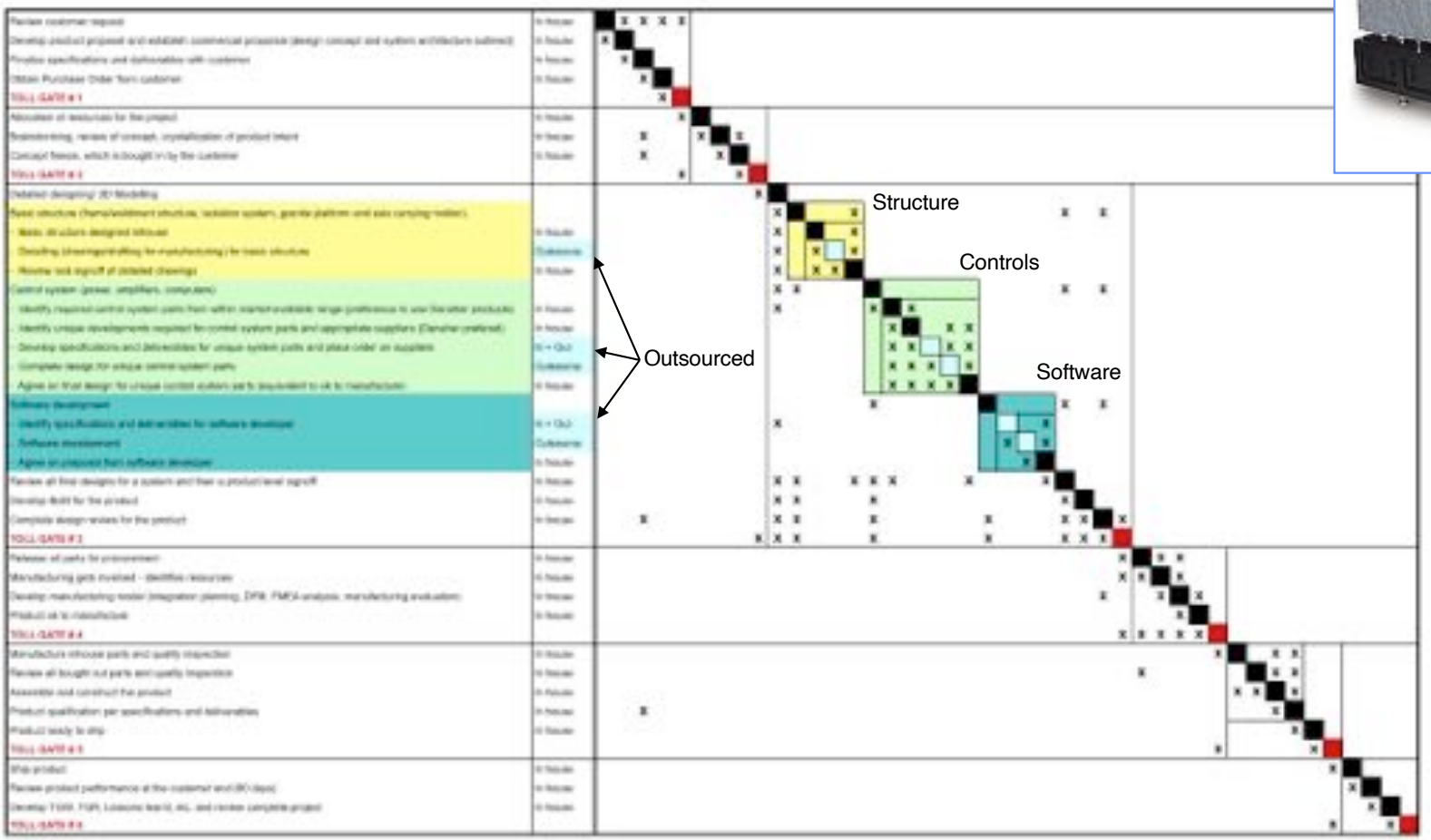
- Marketing
- Engineering
- Finance
- Legal
- Project Management

Interactions

- Across Functions
- Within Functions

Jones Lang LaSalle

Engineering Process DSM: Danaher Motion



Lesson learned: Distributing (outsourcing) coupled tasks is difficult

DANAHER MOTION

How to Create a Process DSM

1. Select a process or sub-process to model.
2. Identify the tasks of the process, who is responsible for each one, and the outputs created by each task.
3. Lay out the square matrix with the tasks in the order they are normally executed.
4. Ask the process experts what inputs are normally used for each task. Insert marks representing the information inputs to each task.
5. Identify the exceptional (unplanned) process flows and the ways that the process can fail. Include marks representing these unplanned iterations.
6. Optional: Analyze the DSM model to re-sequence some of the tasks, suggesting a new process.
7. Optional: Draw solid boxes around the coupled tasks representing the planned iterations.
8. Optional: Draw dashed boxes around groups of parallel (uncoupled) tasks.
9. Highlight key features, such as unplanned iterations.

Two Types of Iteration

Planned Iteration

- Caused by needs to “get it right the first time.”
- We know where these iterations occur, but not necessarily how much.
- Planned iterations should be **facilitated** by good design methods, tools, and coordination.

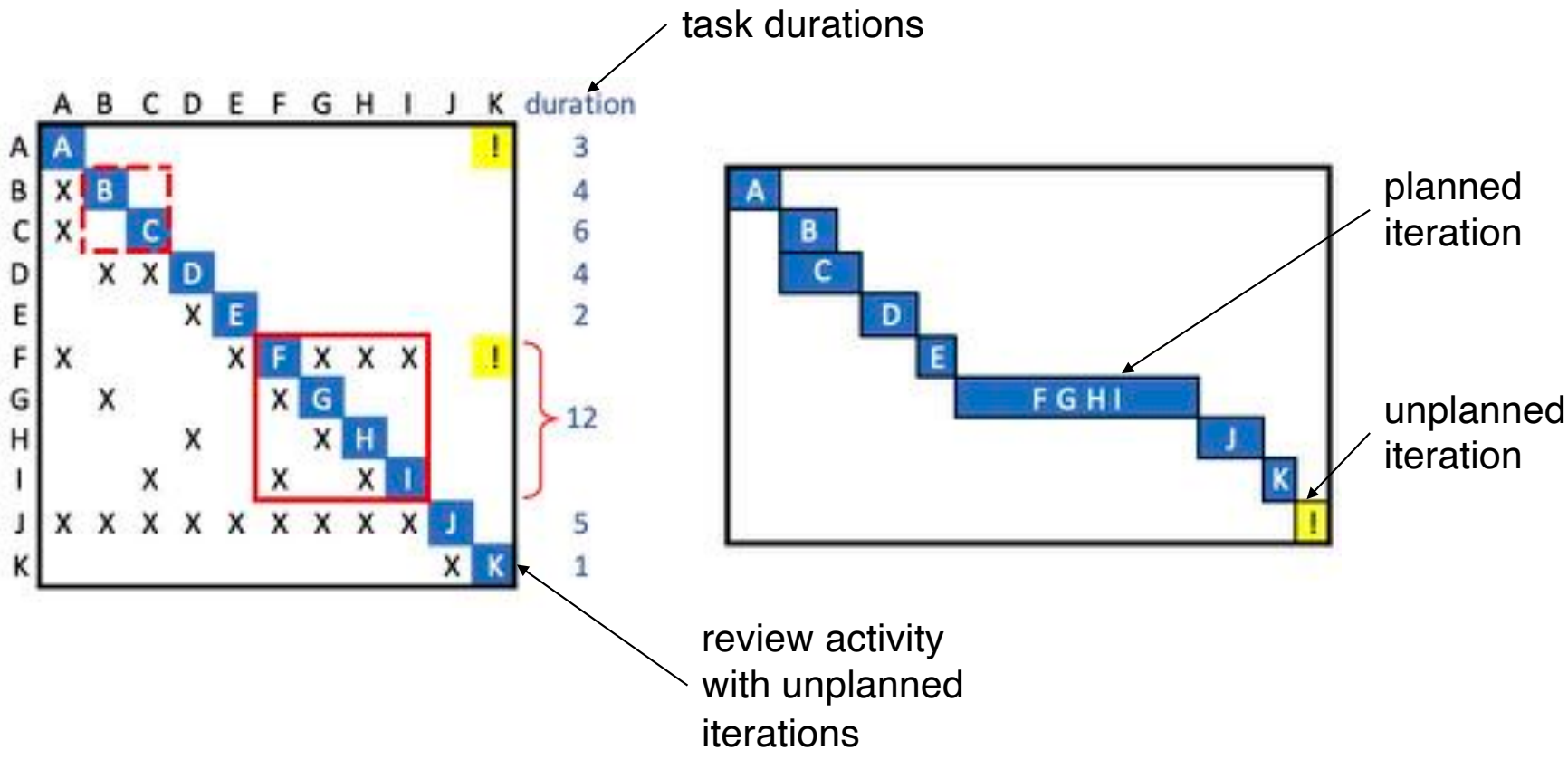
Unplanned Iteration

- Caused by errors and/or unforeseen problems.
- We generally cannot predict which unplanned iterations will occur.
- Unplanned iterations should be **minimized** using risk management methods.

Observation

Mature processes have more planned iterations and fewer unplanned iterations.

Converting a Process DSM to a Gantt Chart



DSM Building Exercise

Convert process diagram into DSM

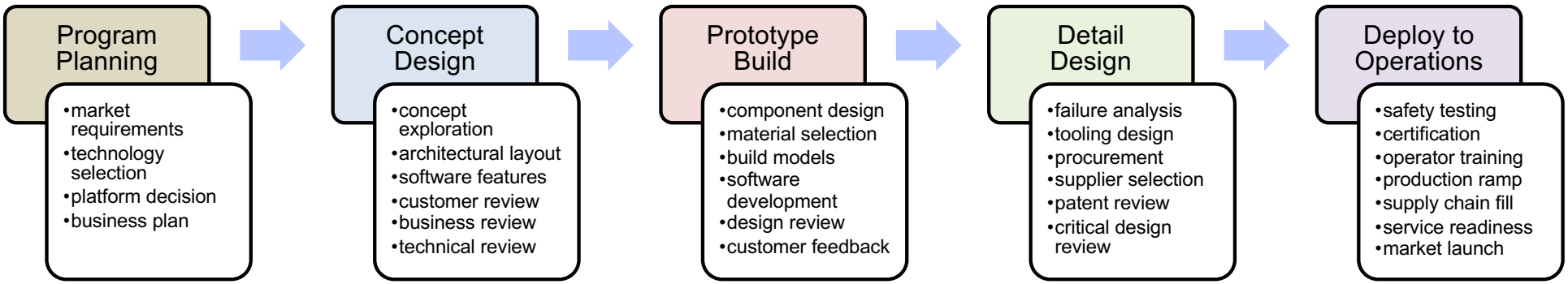
- Lay out the phases and tasks within each.
- Insert two or more input marks for each task.
- Identify the planned and unplanned iterations.
- Add more tasks if helpful for completeness.

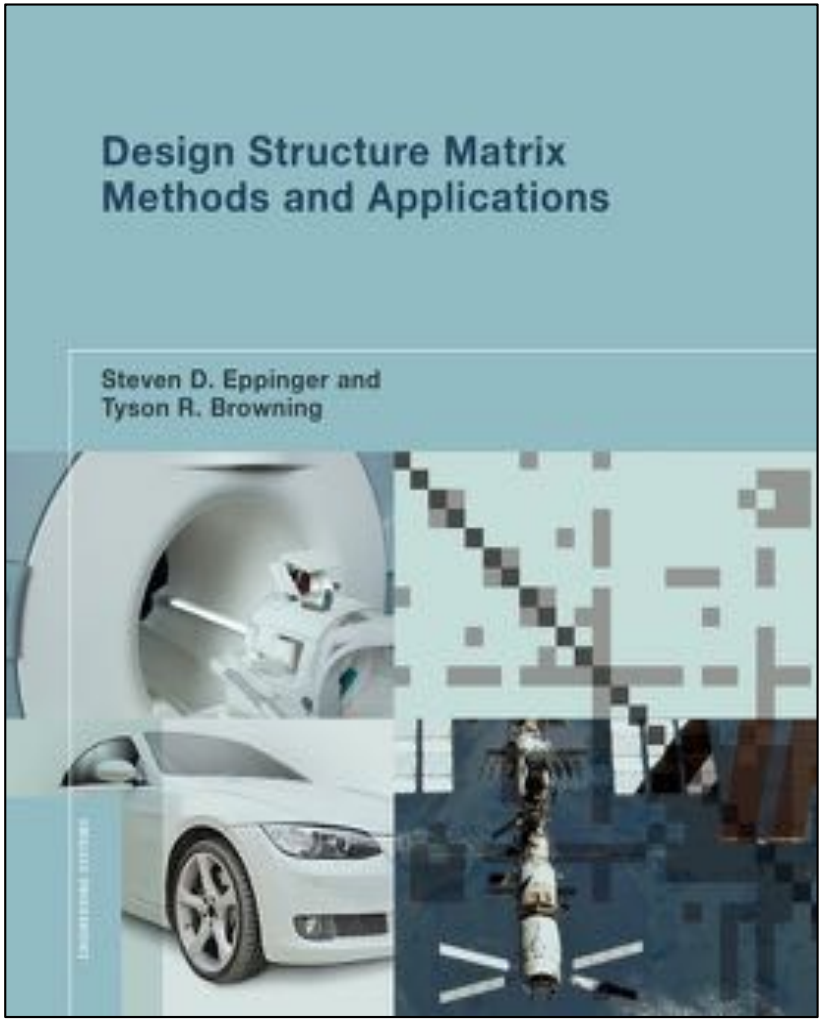
Challenge

- Add the testing tasks where they should be.

Discussion

- What can you learn from such a DSM model?





- 80 contributors
- 44 DSM examples
- 200 color illustrations
- 330 pages

Steven D. Eppinger and Tyson R. Browning
Design Structure Matrix Methods and Applications
MIT Press, Cambridge, 2012.

Design Structure Matrix Methods and Applications

① Introduction to DSM Methods

② Product Architecture DSM Models

③ Product Architecture DSM Examples

④ Organization Architecture DSM Models

⑤ Organization Architecture DSM Examples

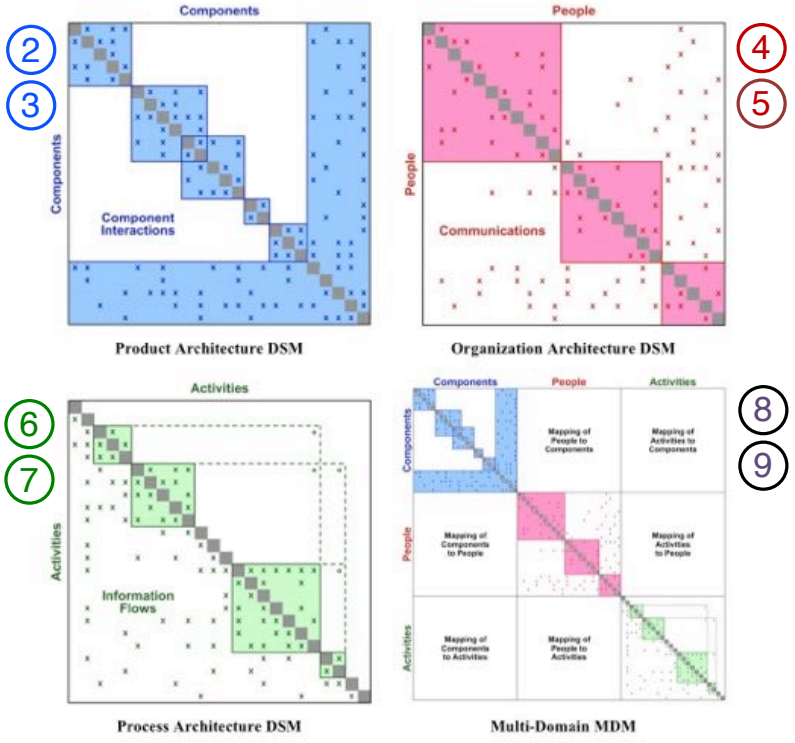
⑥ Process Architecture DSM Models

⑦ Process Architecture DSM Examples

⑧ Multi-Domain Architecture MDM Models

⑨ Multi-Domain Architecture MDM Examples

⑩ The Future of DSM



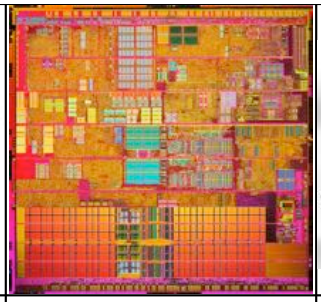
Product Architecture DSM

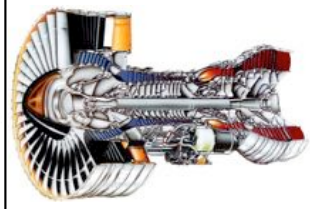
Organization Architecture DSM

Process Architecture DSM

Multi-Domain MDM

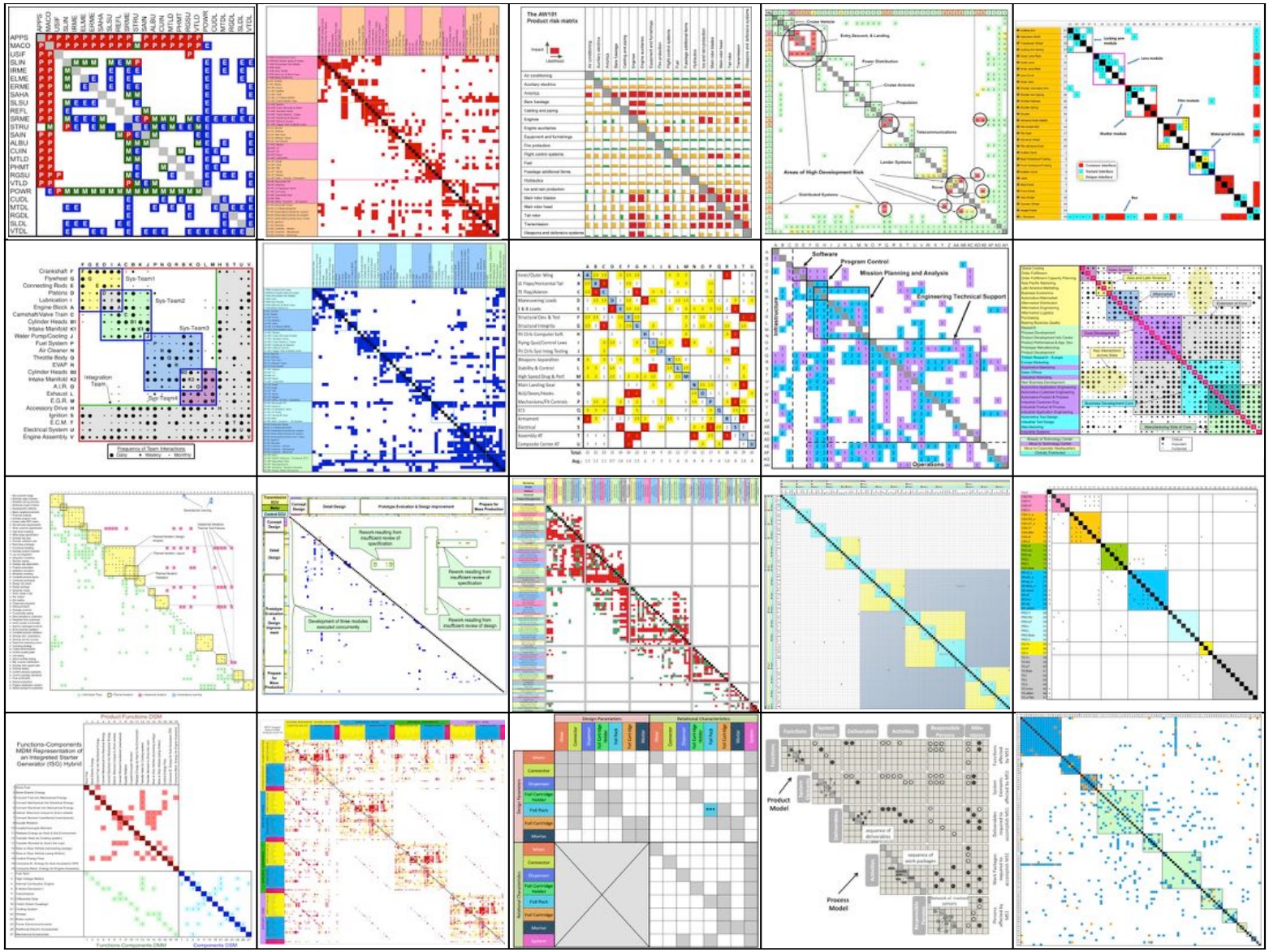

Kodak






Exp 12/02
162000-1


HILTI


APPS
MACO
USIF
SLIN
IRME
ELME
ERME
SAHA
SLSU
REFL
SRME
STRU
SAIN
ALBU
CUIN
MTLD
PHMT
RGSU
VTLD
POWR
CUDL
MTDL
RGDL
SLDL
VTDL
The AW101 Product risk matrix
Impact
Likelihood
Air conditioning
Auxiliary electrics
Avionics
Bare fuselage
Cabling and piping
Engines
Engine auxiliaries
Equipment and furnishings
Fire protection
Flight control systems
Fuel
Fuselage additional items
Hydraulics
Ice and rain protection
Main rotor blades
Main rotor head
Tail rotor
Transmission
Weapons and defensive systems
Cruise Vehicle
Entry,Descent, & Landing
Power Distribution
Cruise Avionics
Propulsion
Telecommunications
Lander Systems
Areas of High Development Risk
Rover
Distributed Systems
Locking arm module
Lens module
Filter module
Shutter module
Waterproof module
Bus
Crankshaft
Flywheel
Connecting Rods
Pistons
Lubrication
Engine Block
Camshaft/Valve Train
Cylinder Heads
Intake Manifold
Water Pump/Cooling
Fuel System
Air Cleaner
Throttle Body
EVAP
Cylinder Heads
Intake Manifold
A.I.R.
Exhaust
E.G.R.
Accessory Drive
Ignition
E.C.M.
Electrical System
Engine Assembly
Sys-Team1
Sys-Team2
Sys-Team3
Integration Team
Sys-Teams
Frequency of Team Interactions
Daily
Weekly
Months
Software
Program Control
Mission Planning and Analysis
Engineering Technical Support
Infrastructure
Operations
Rework resulting from insufficient review of specification
Rework resulting from insufficient review of specification
Development of three modules executed concurrently
Rework resulting from insufficient review of design
Functions-Components MDM Representation of an Integrated Starter Generator (ISG) Hybrid
Product-Functions DSM
Functions-Components DSM
Components DSM
Design Parameters
Relational Characteristics
Product Model
Process Model
sequence of deliverables
sequence of work packages

Managing Complex Technical Projects

Day 2: System Architecture and Organization DSM Models

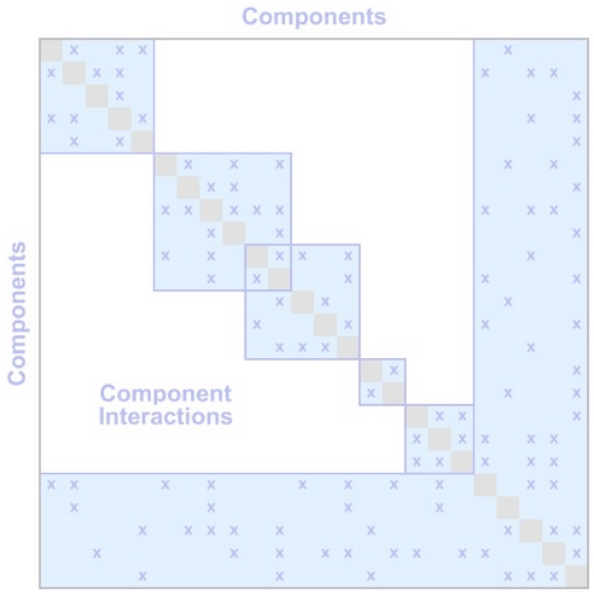
Prof. Steven D. Eppinger

Massachusetts Institute of Technology

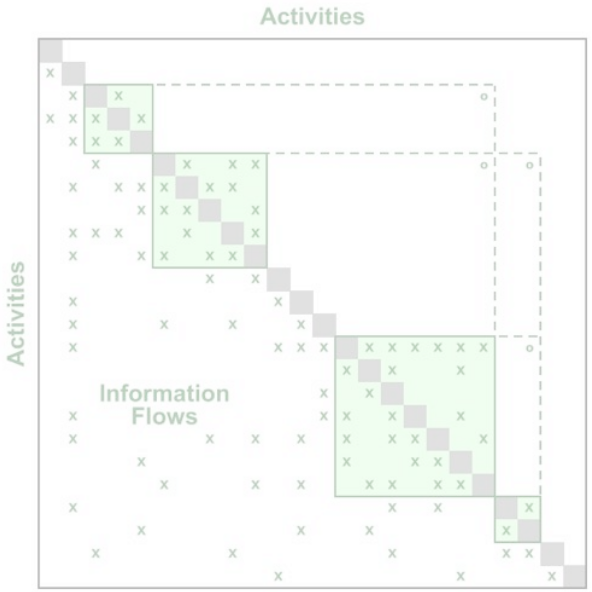
Sloan School of Management

eppinger@mit.edu
web.mit.edu/eppinger
www.dsmweb.org

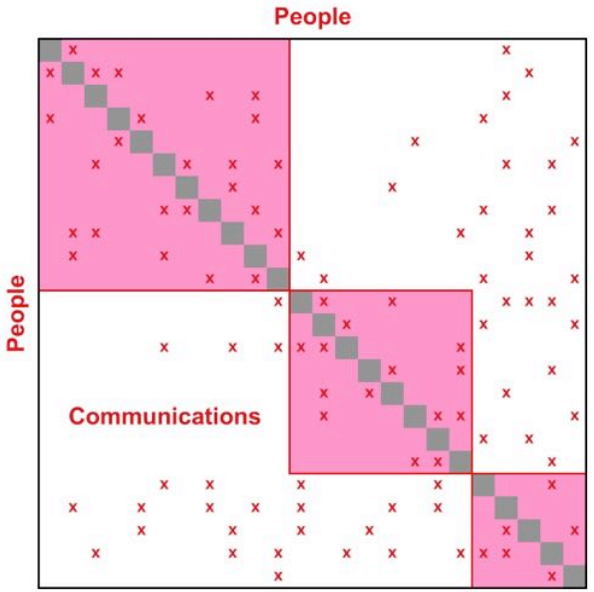
Three Primary DSM Types



Product Architecture
DSM



Process Architecture
DSM

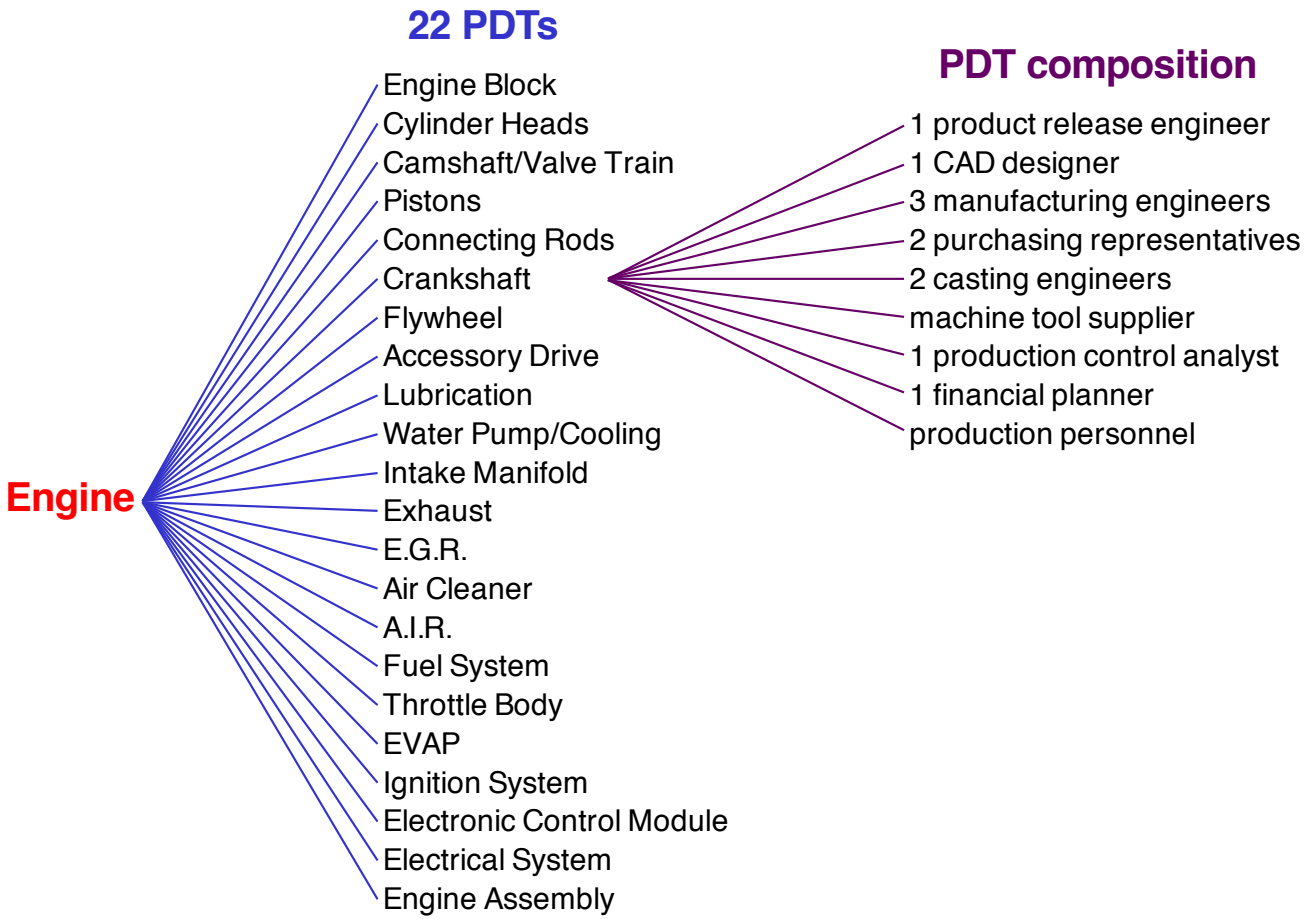


Organization Architecture
DSM

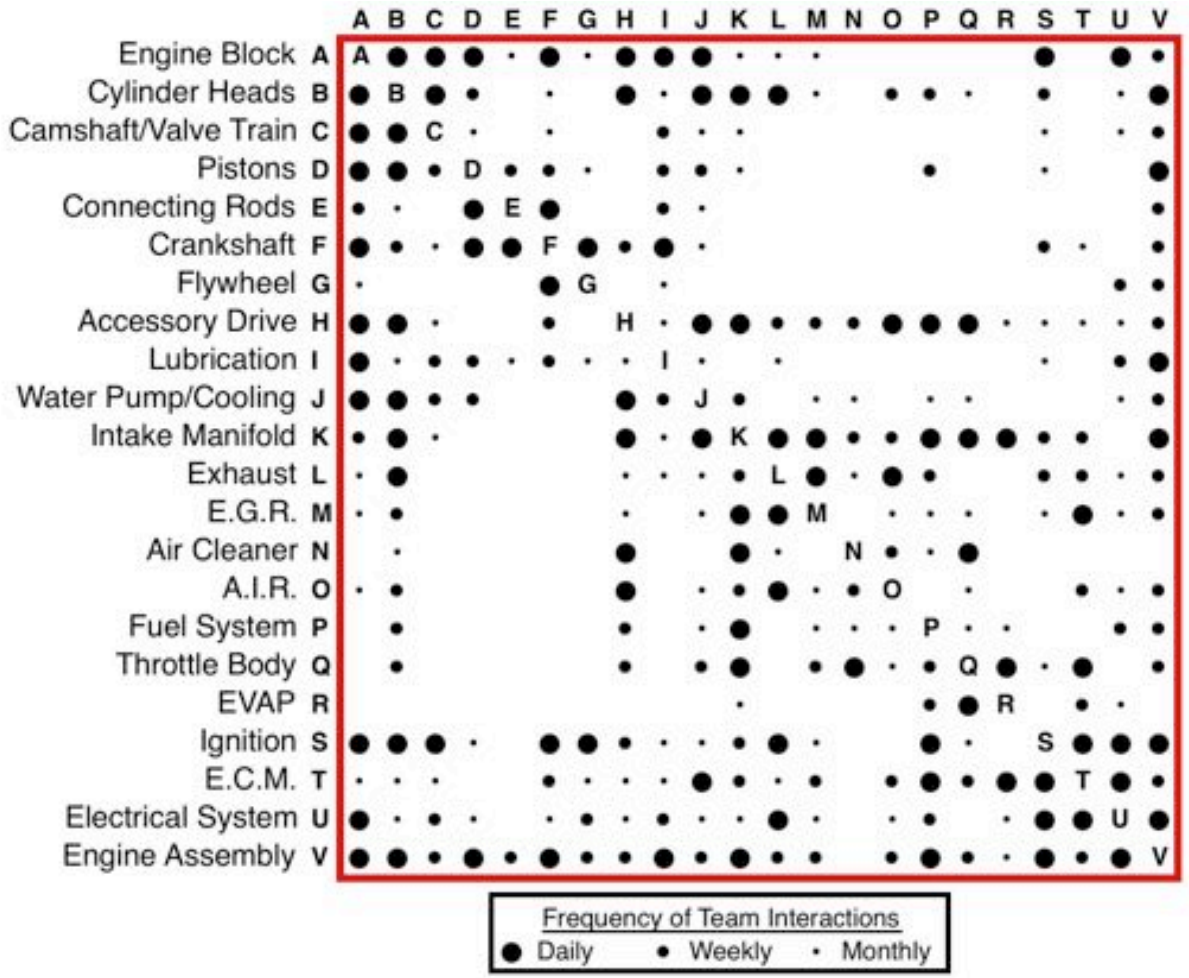
Organization DSM Application: Engine Development

- Site: General Motors Powertrain Division
- Product: “new-generation” engine
- Structure: 22 PDTs involved simultaneously

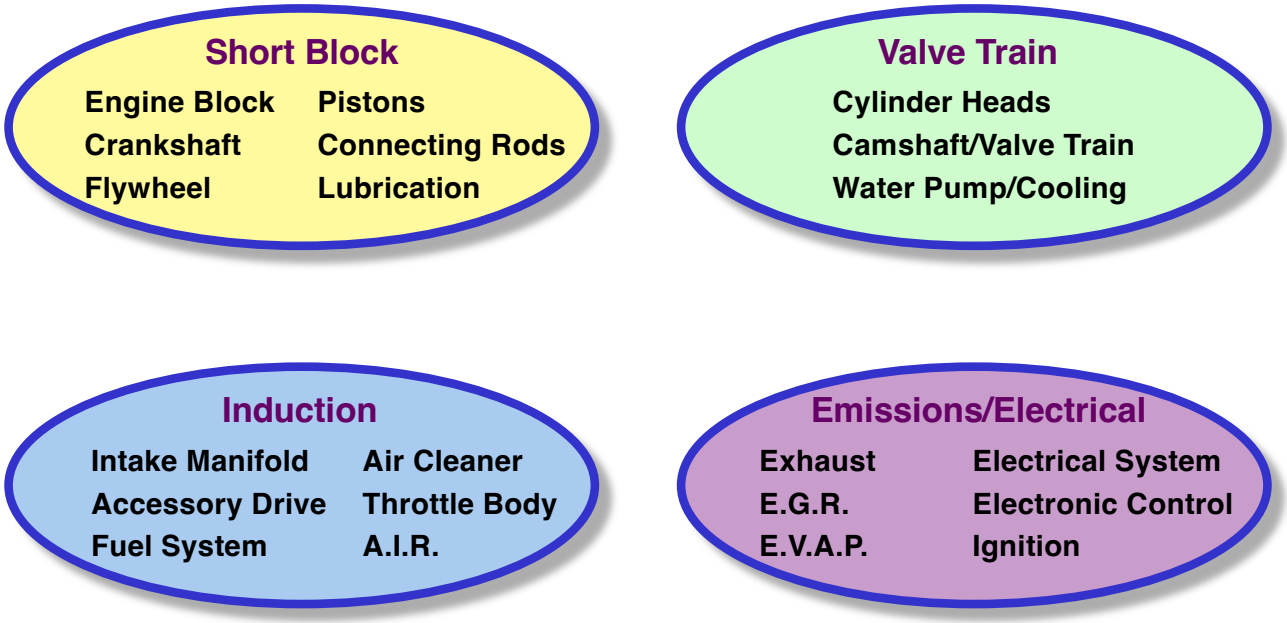
Decomposition of the Engine Development Project



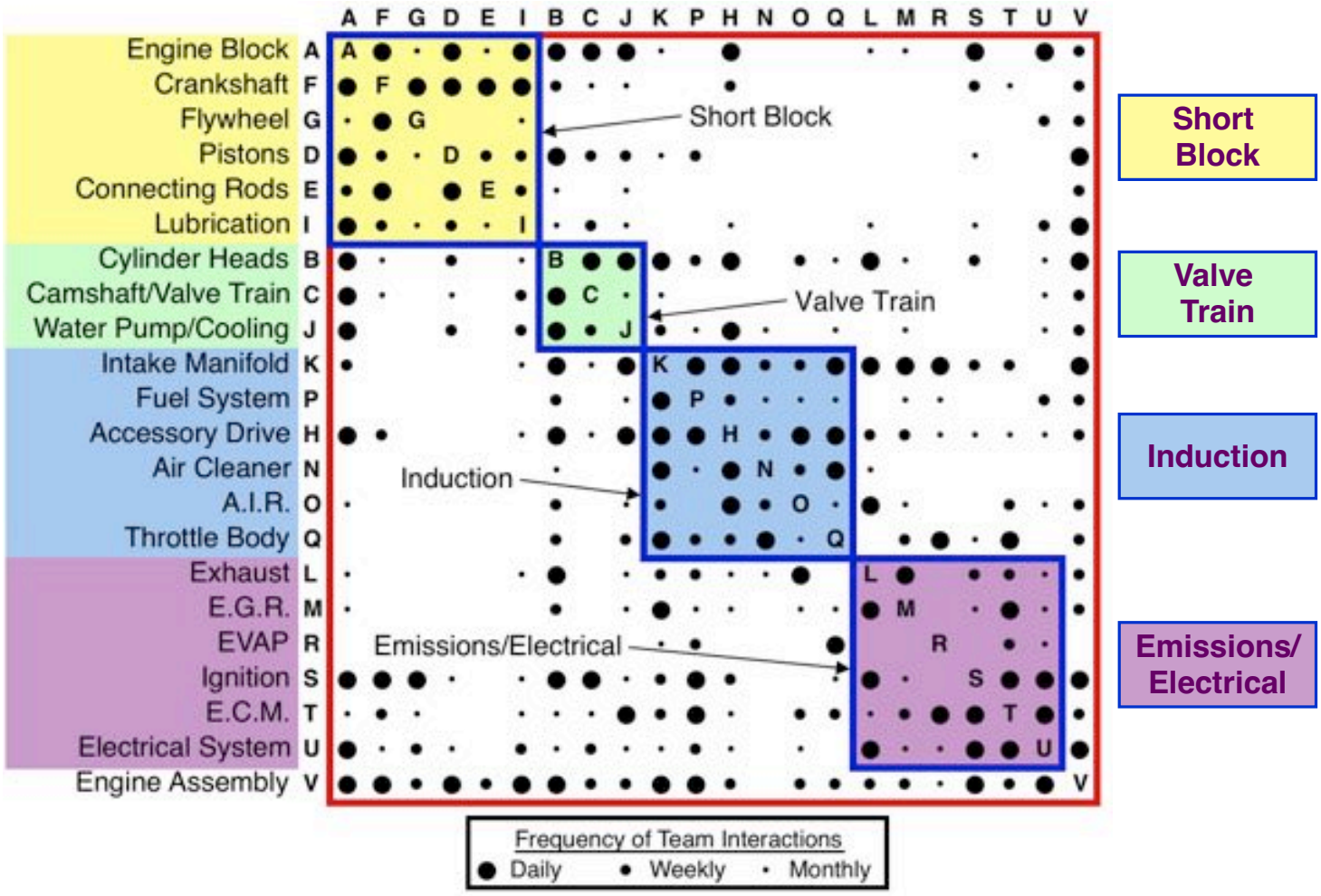
PDT Interactions



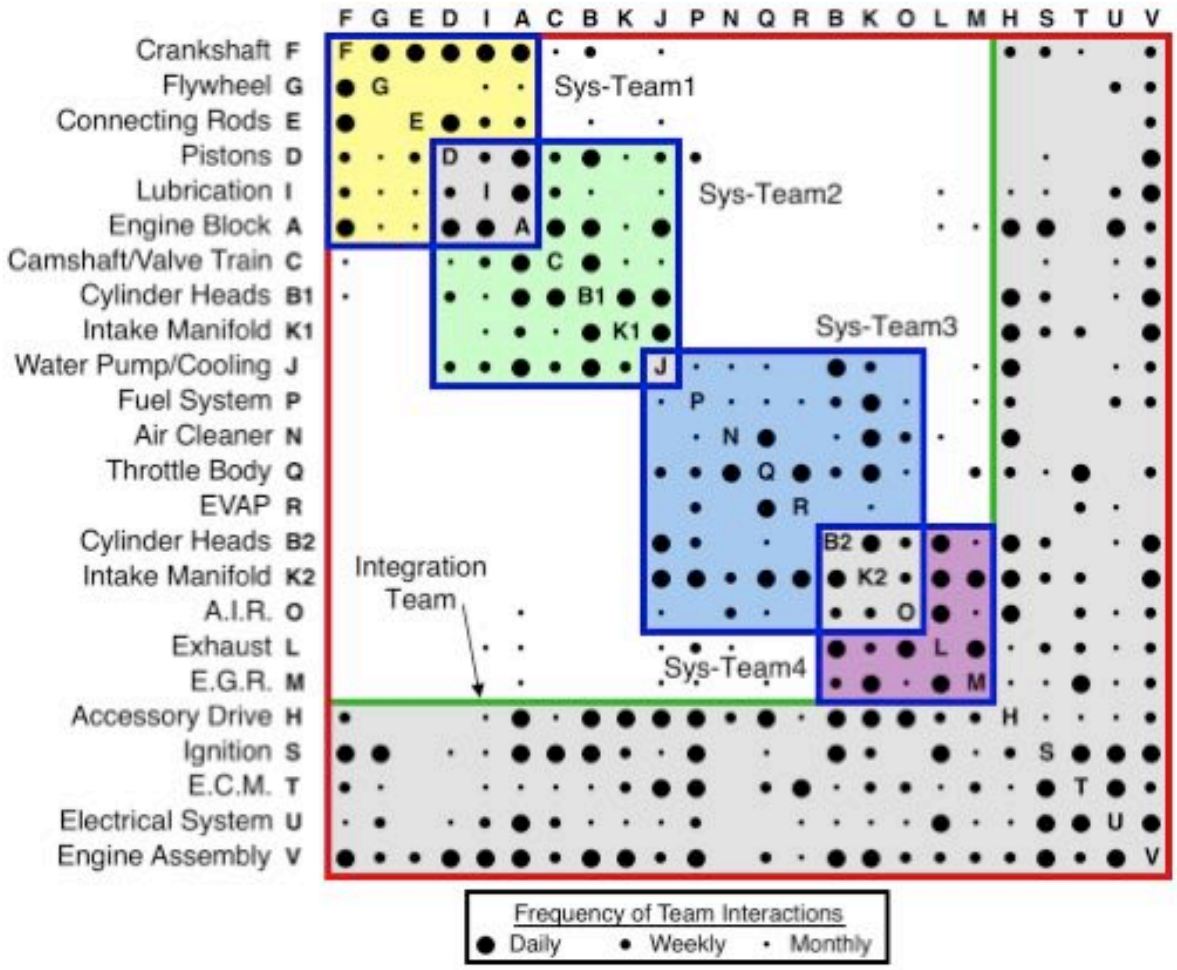
System Team Assignments

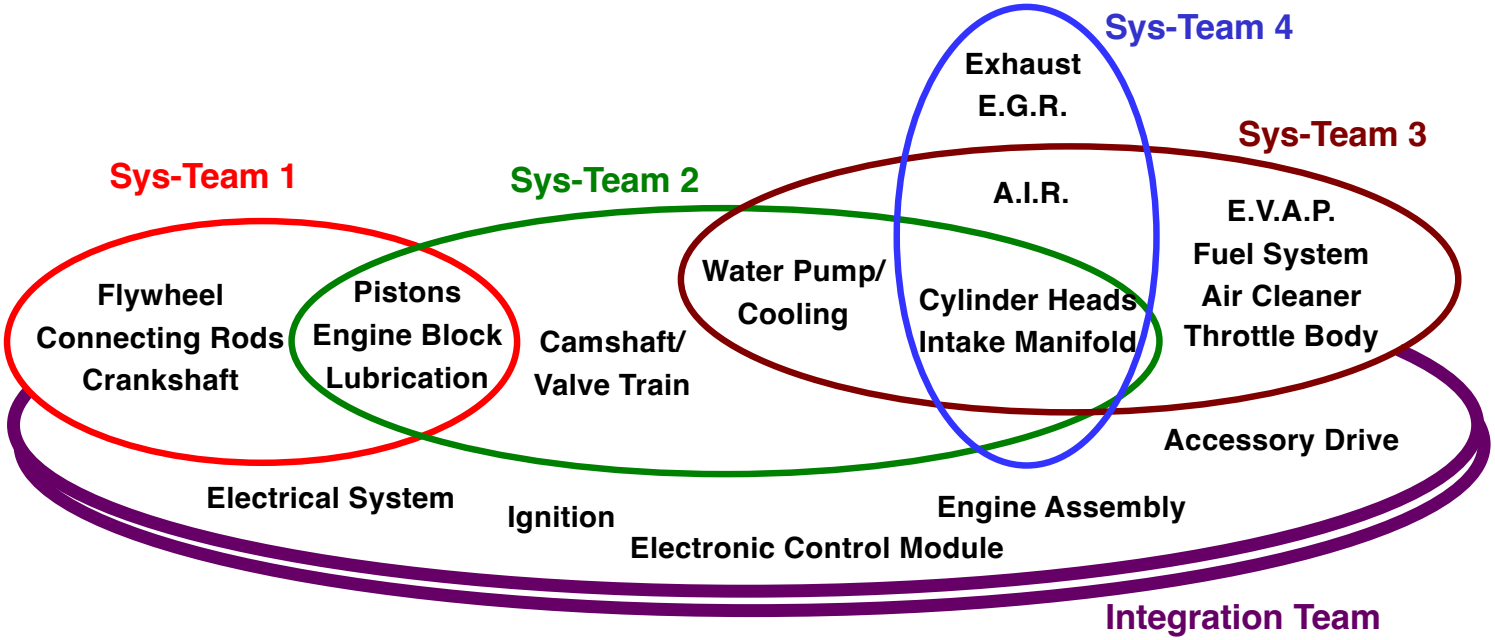


Existing System Teams



Proposed System Teams



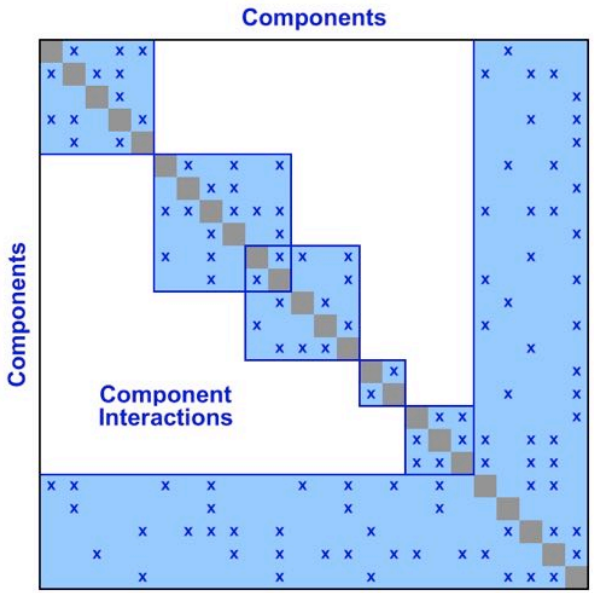


PDT-to-System-Team Assignments

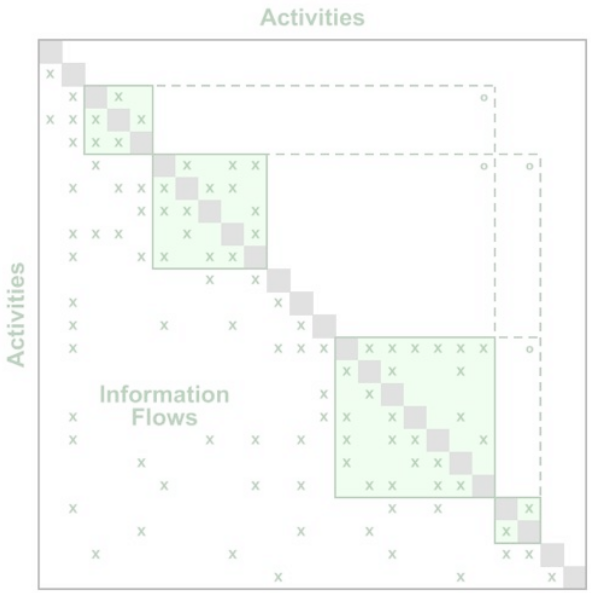
How to Create an Organization DSM

1. Select a unit of analysis for the elements of the DSM (e.g., individuals, job roles, teams, departments, suppliers, or other entities).
2. List all of these “people entities” in the rows and columns of the DSM.
3. Decide what types of relationships to represent (e.g., work-related communications, social connections between people or organizational units).
4. Collect the data. Ask people about these relationships or use another source of data.
5. (Optional) Group the people according to the existing organization structure.
6. (Optional) Use clustering analysis to suggest a new structure. Manual clustering or automatic algorithms can be used.
7. (Optional) Augment the DSM with any graphics that are helpful to understand and communicate the insights you get from the DSM results.

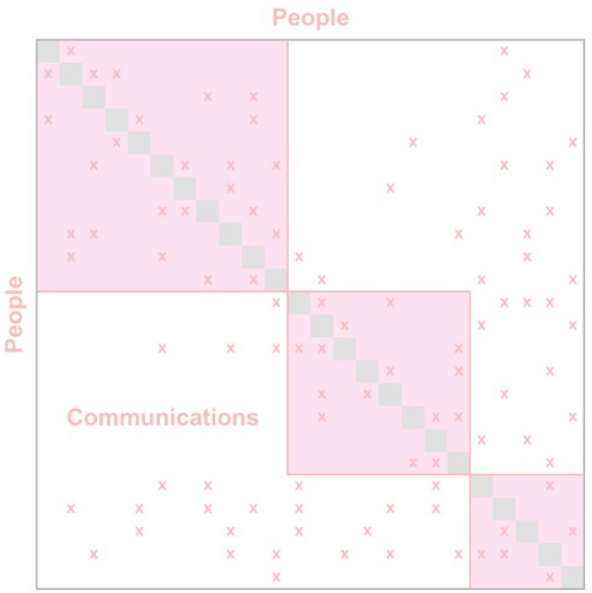
Three Primary DSM Types



System ~~Product~~ Architecture DSM



Process Architecture DSM



Organization Architecture DSM

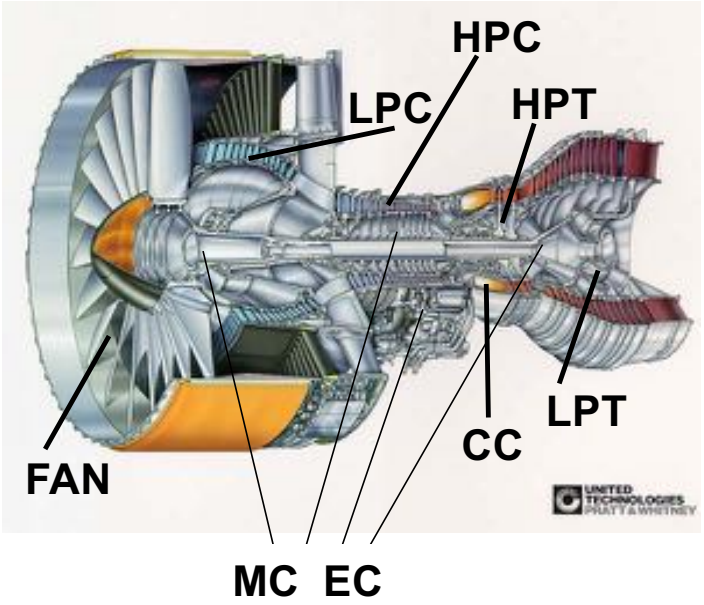
System Architecture DSM: P&W 4098 Jet Engine

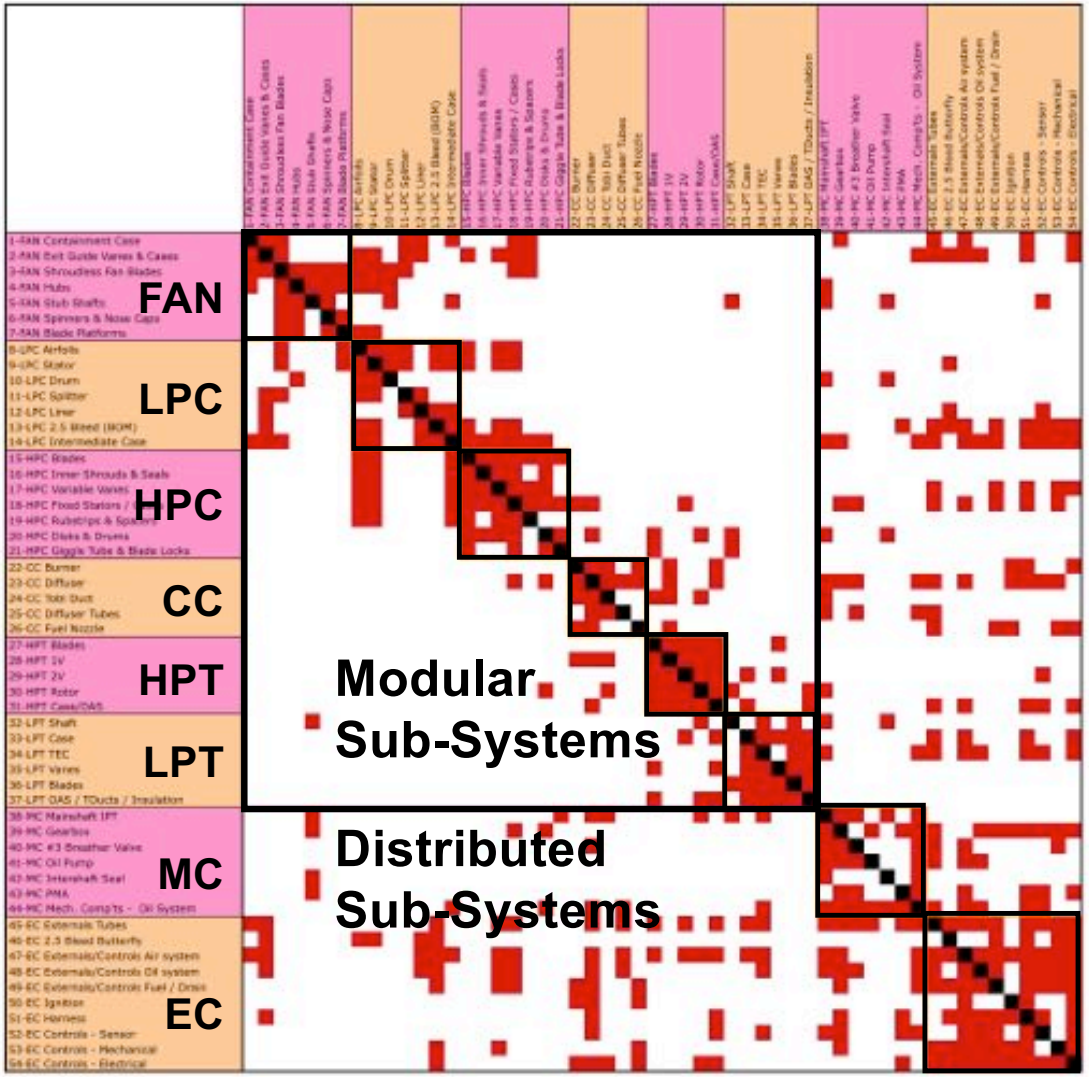
8 Sub-Systems

54 Components

569 Interfaces

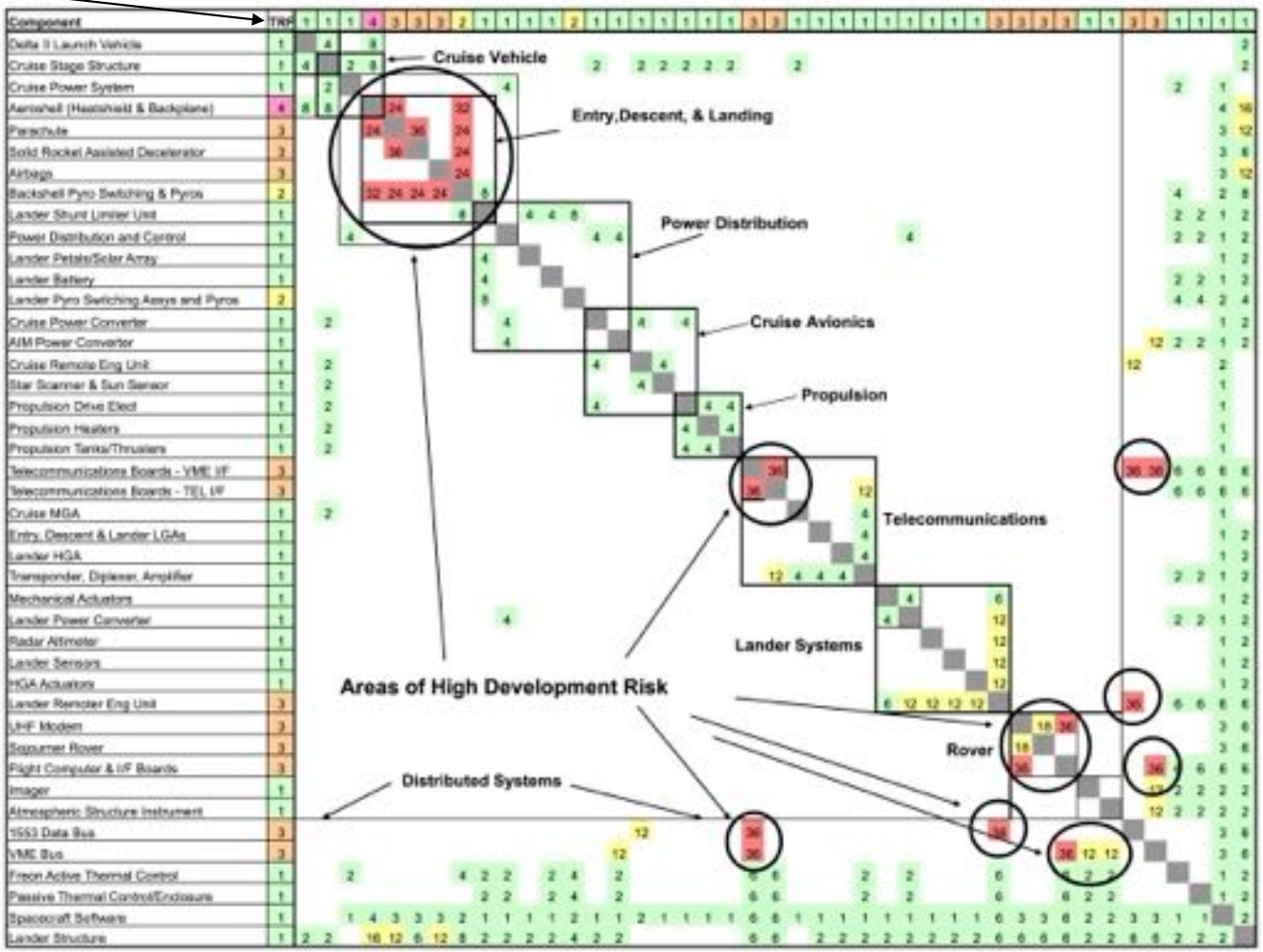
Interface Types
- Spatial
- Structural
- Energy
- Materials
- Data/Controls





Mars Pathfinder Technology Risk DSM

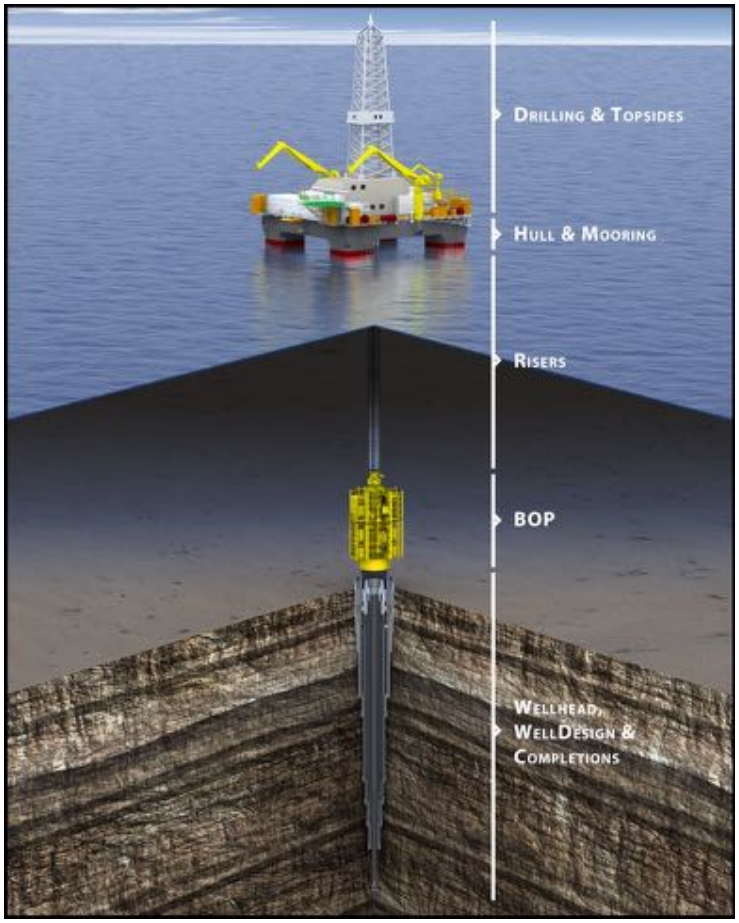
Component-level technology risk factor (TRF) based on each component's technology readiness level (TRL)



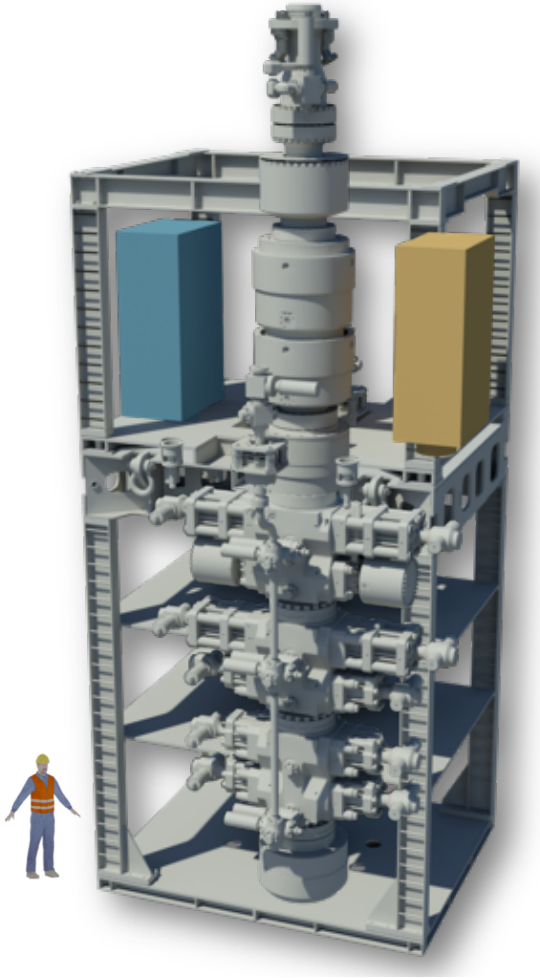
TR DSM value (i,j) = DSM(i,j) x TRF(i) x TRF(j)

Ref: Brady, TK, "Utilization of Dependency Structure Matrix Analysis to Assess Complex Project Designs," *ASME Design Engineering Technical Conferences*, Montreal, 2002 .

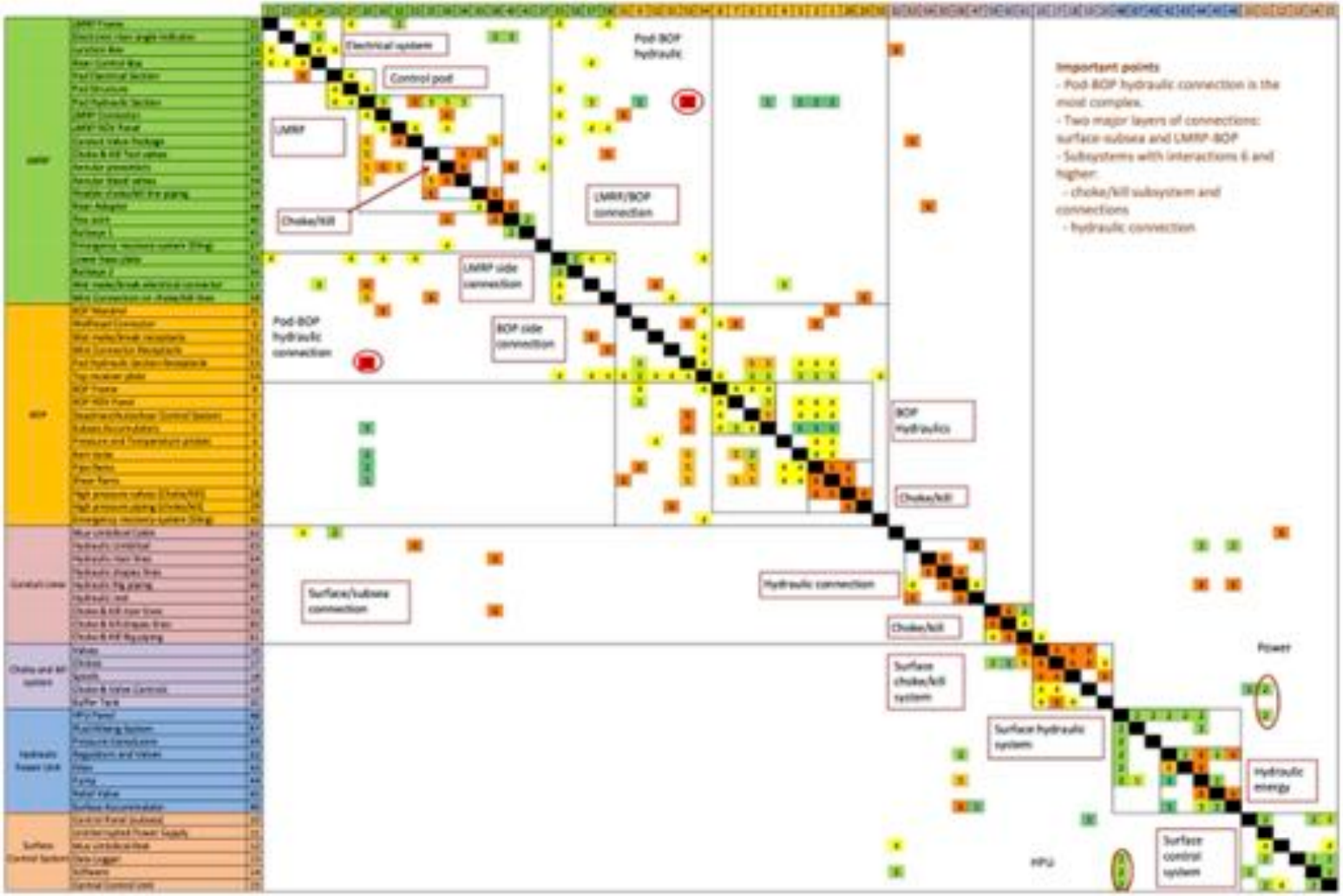
Deepwater Oil & Gas Drilling Operation



Blowout Preventer (BOP)



Blowout Preventer System Architecture DSM

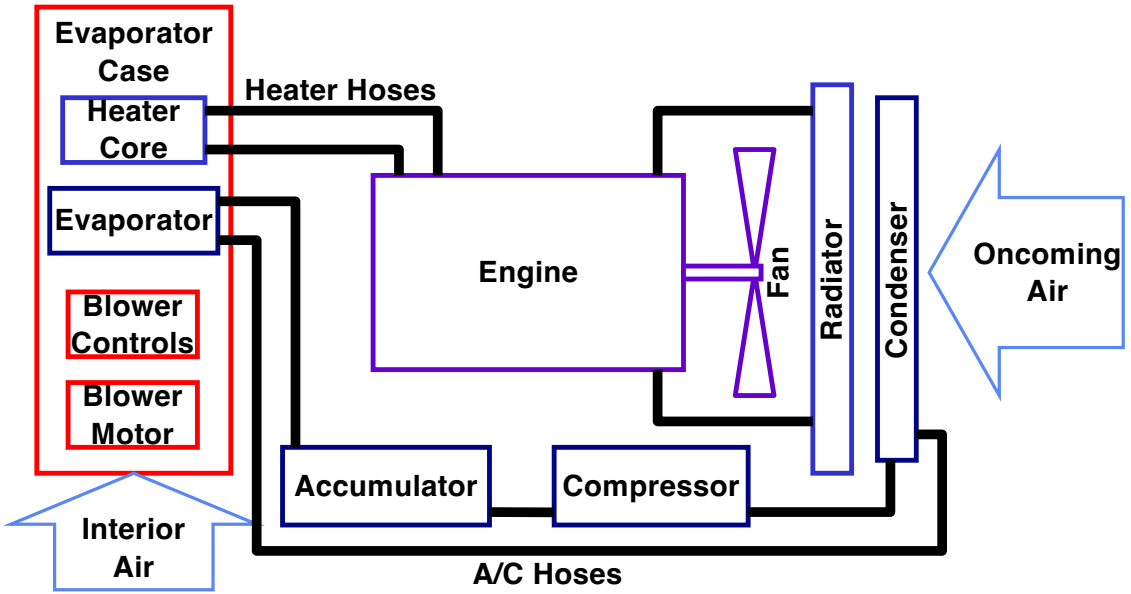


How to Create a System Architecture DSM

1. Start with the decomposition of the system into sub-systems and components. List these elements in the rows and columns of the DSM.
2. Select the type(s) of interfaces to be represented in the DSM (e.g., physical, spatial, energy, electrical, hydraulic, information, signal, types of material flows)
3. Ask people who know about the components to identify all the interfaces for each component. Represent the interfaces using (one or more types of) marks in the DSM.
4. Cluster the DSM according to the existing sub-system structure.
5. (Optional) Re-cluster the components to suggest potentially new groupings into sub-systems or modules. Manual clustering or automatic algorithms can be used.
6. (Optional) Augment the DSM with any graphics that are helpful to understand and communicate the insights you get from the DSM results.

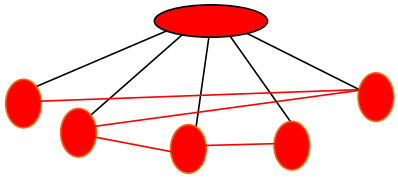
DSM Building Exercise:
Climate Control System Development

How would you organize the development of the various sub-systems to assure proper integration and quality of the overall system?



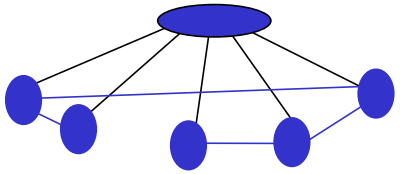
Comparing the System Architecture to the Organization Structure

System Decomposition into Components



Component interfaces define the architecture

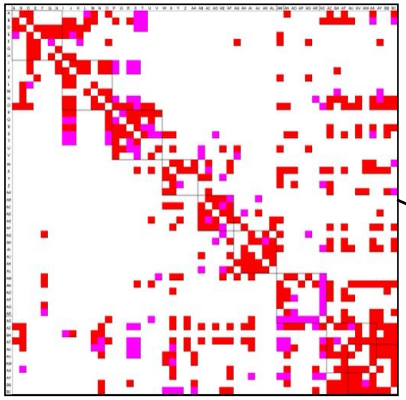
Organization Decomposition into Teams



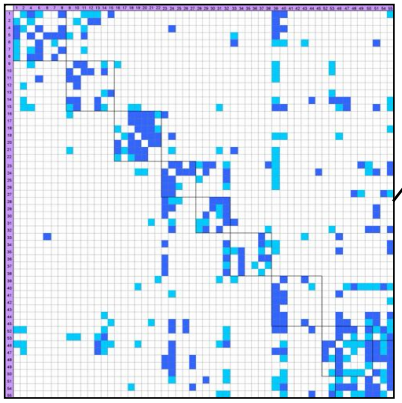
Team interactions implement the architecture

How does system architecture drive development team interaction?

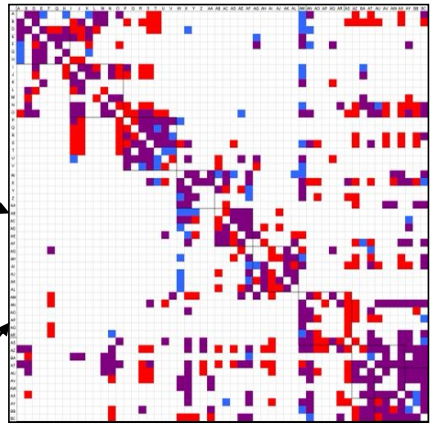
Mapping Component Interfaces to Team Interactions using DSM



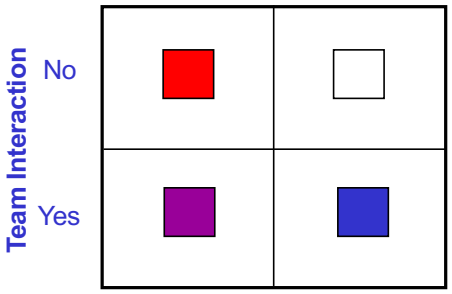
Product Architecture DSM
(Component Interfaces)



Organization Architecture DSM
(Team Interactions)

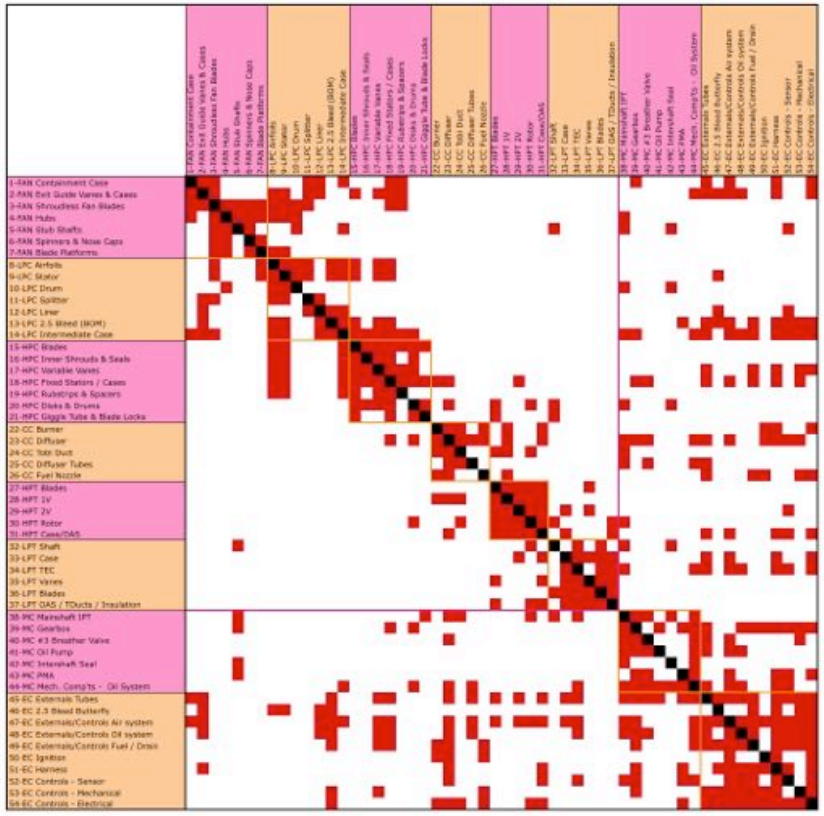


Two-Domain DSM
(Comparison)



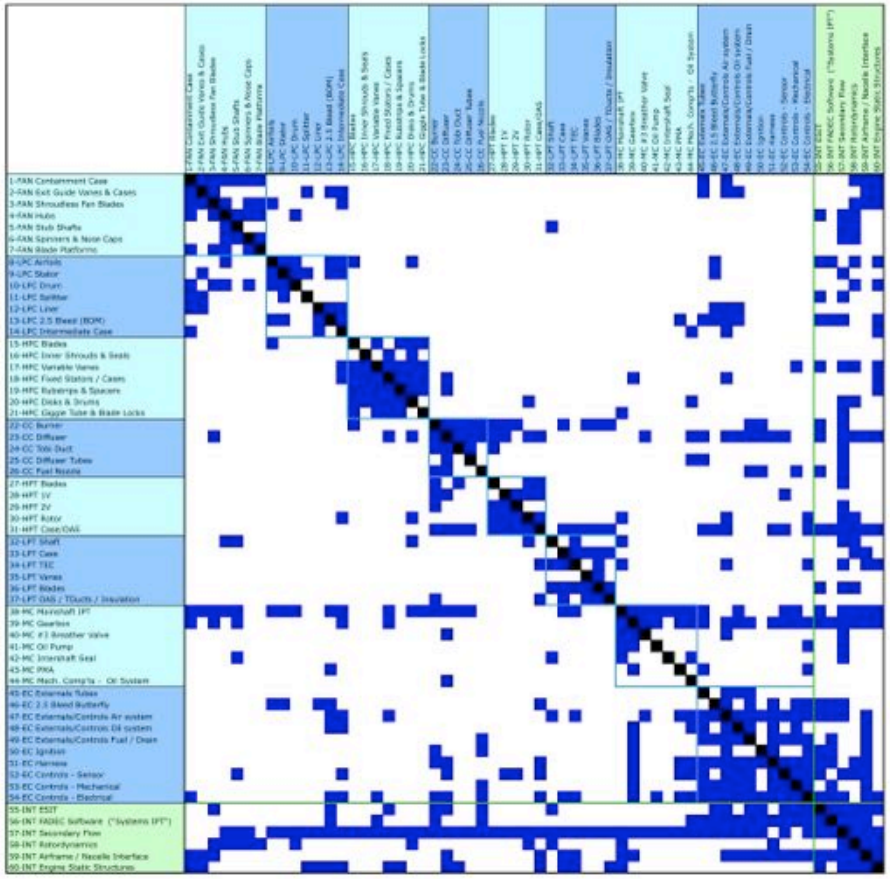
P&W 4098 Jet Engine

Component Interfaces



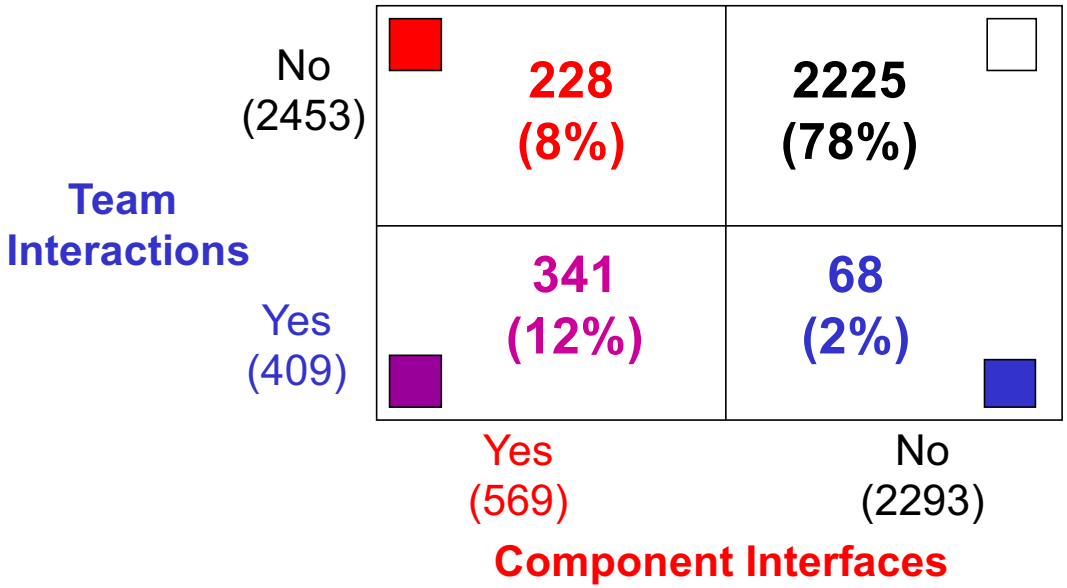
54 components → 8 sub-systems

Team Interactions



54 component teams → 8 module teams
+6 system integration teams

Most Team Interactions Match Component Interfaces

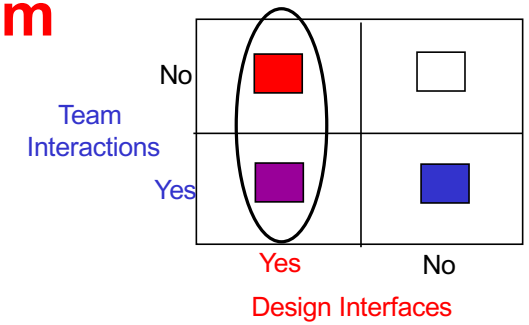


References:

- Sosa, Eppinger, and Rowles, ”Identifying Modular and Integrative Systems and Their Impact on Design Team Interactions”, *Journal of Mechanical Design*, June 2003.
- Sosa, Eppinger, and Rowles, "The Misalignment of Product Architecture and Organizational Structure in Complex Product Development”, *Management Science*, Dec. 2004.

Effect of Organization/System Boundaries

Data set: 569 component interfaces



First criterion:

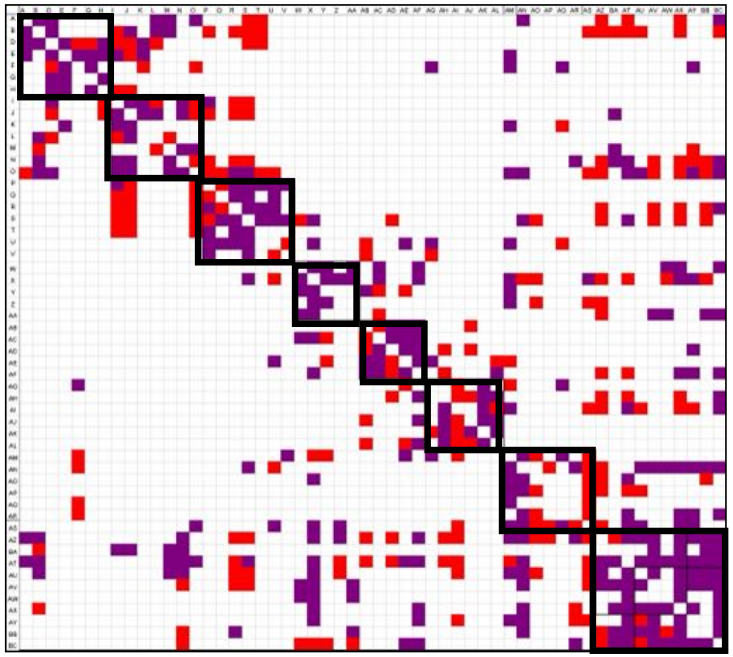
Interfaces matched by team interactions — **59.9%**

Interfaces NOT matched by team interactions — **40.1%**

Second criterion:

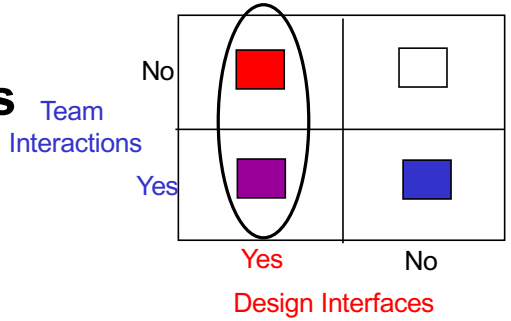
Interfaces WITHIN organizational boundaries — 78.8% are matched

Interfaces ACROSS organizational boundaries — 47.8% are matched



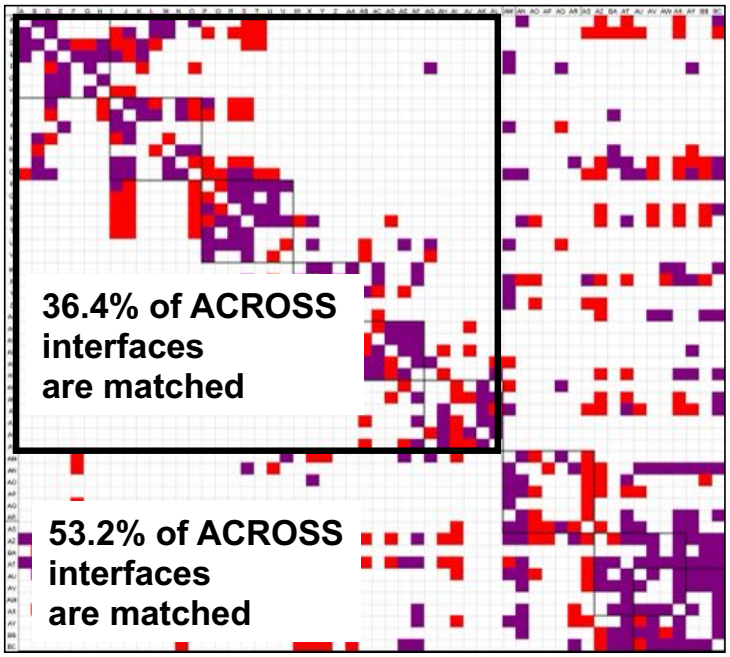
Organizational/System Boundaries:
Modular vs. Distributed Sub-Systems

Data set: 569 design interfaces

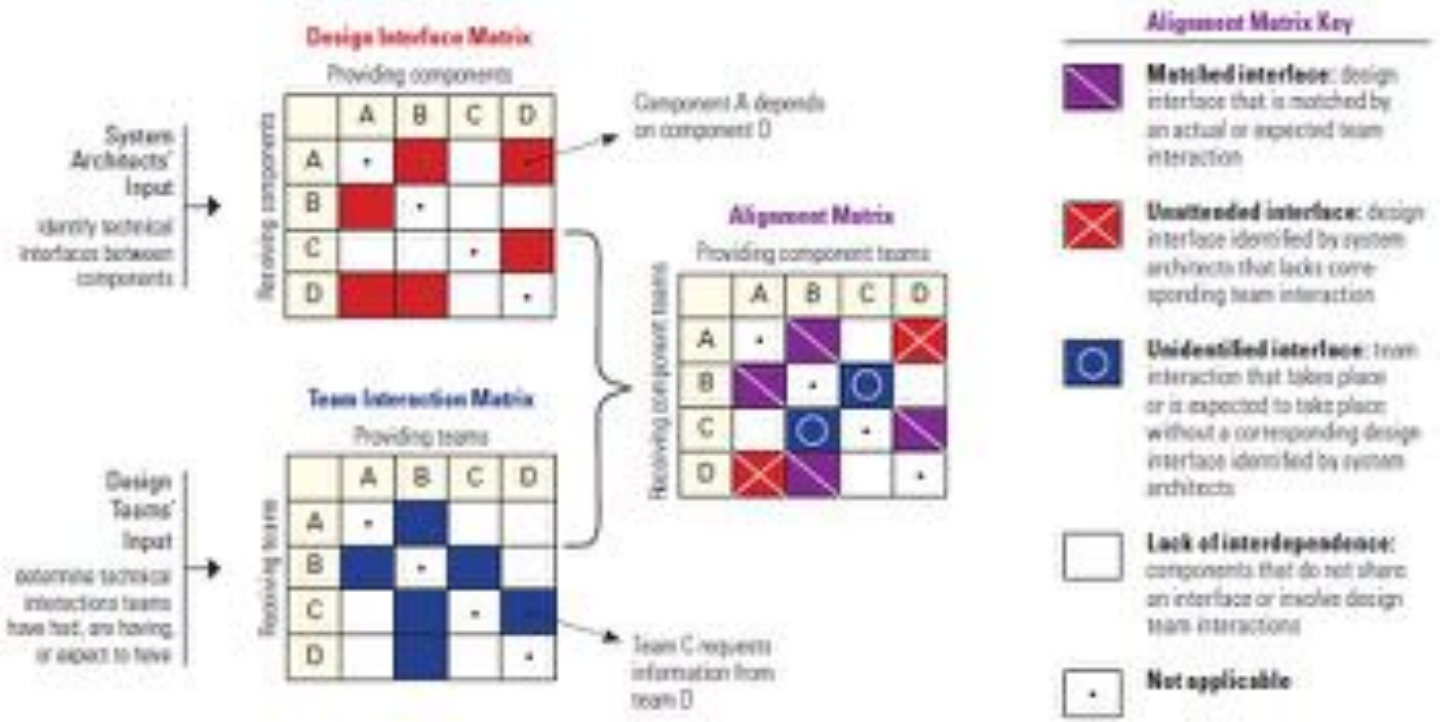


Overall:

Interfaces WITHIN organizational boundaries	78.8% are matched
Interfaces ACROSS organizational boundaries	**47.8% are matched**



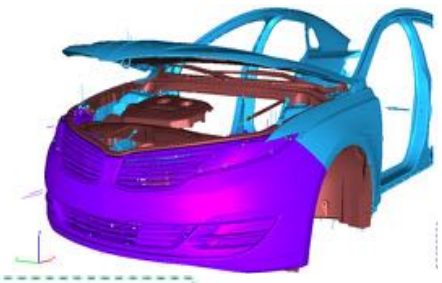
Simultaneous Product and Organization Architecture DSM Clustering

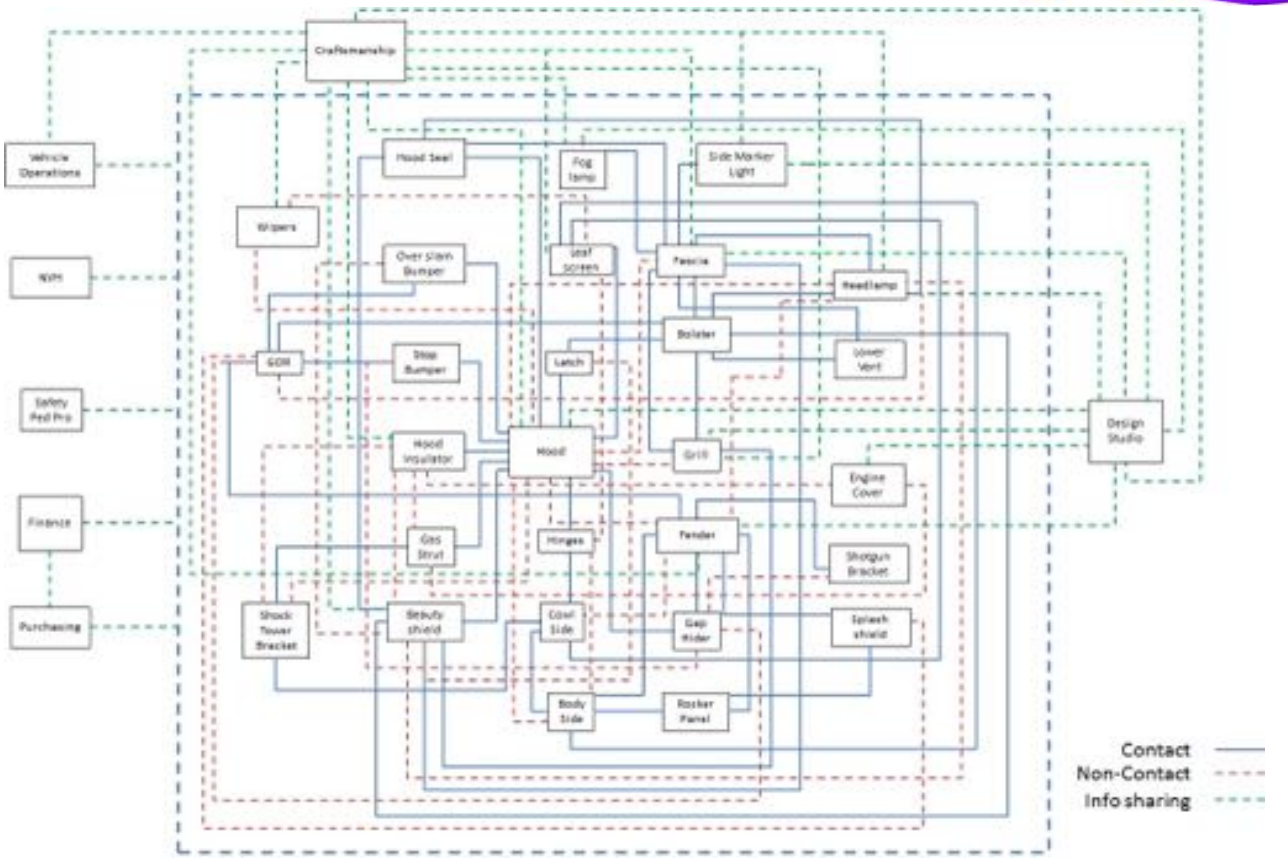


Reference:

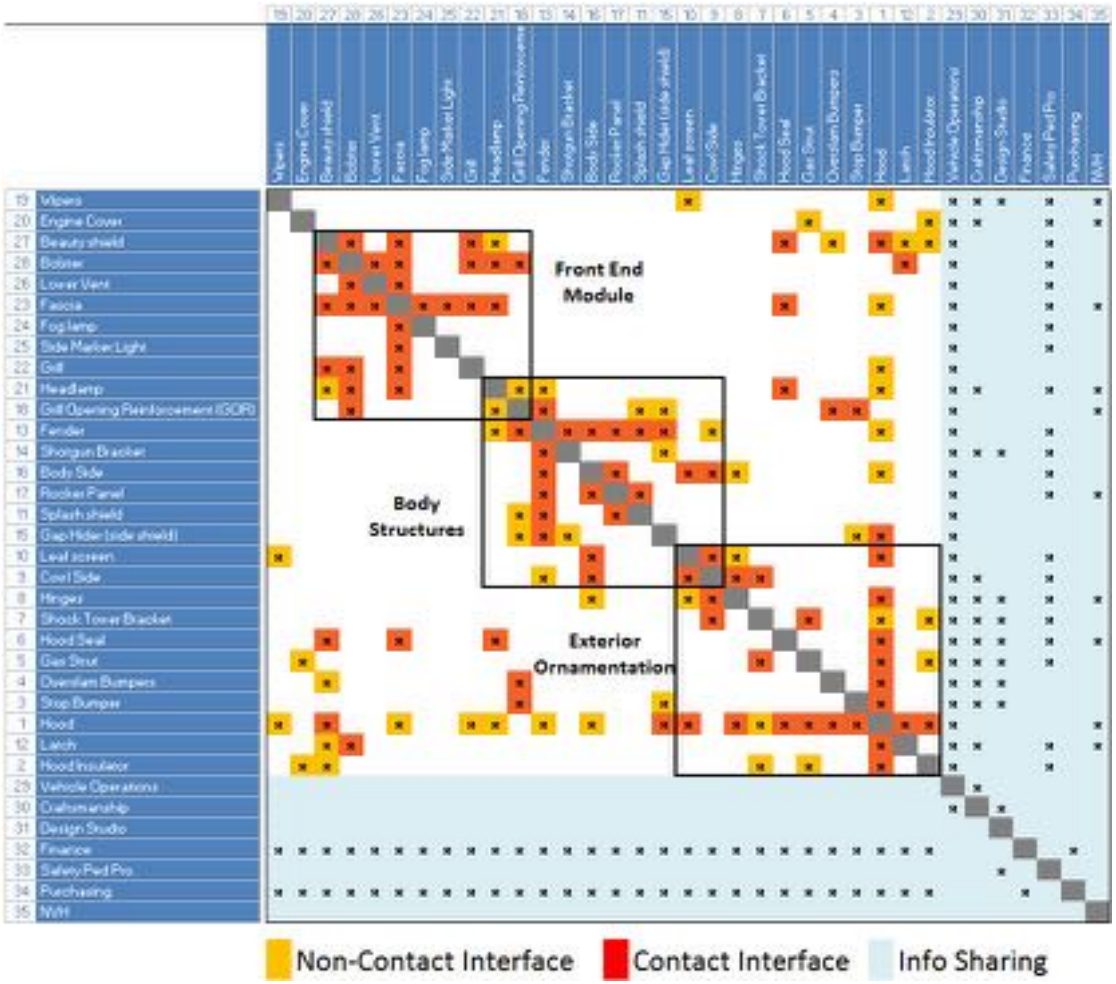
- R. Reyes and S. Eppinger, " Designing a Product Development Organization Based on the Product Architecture", *17th International DSM Conference*, Ft. Worth, November 2015.

Vehicle Front-End Product Architecture

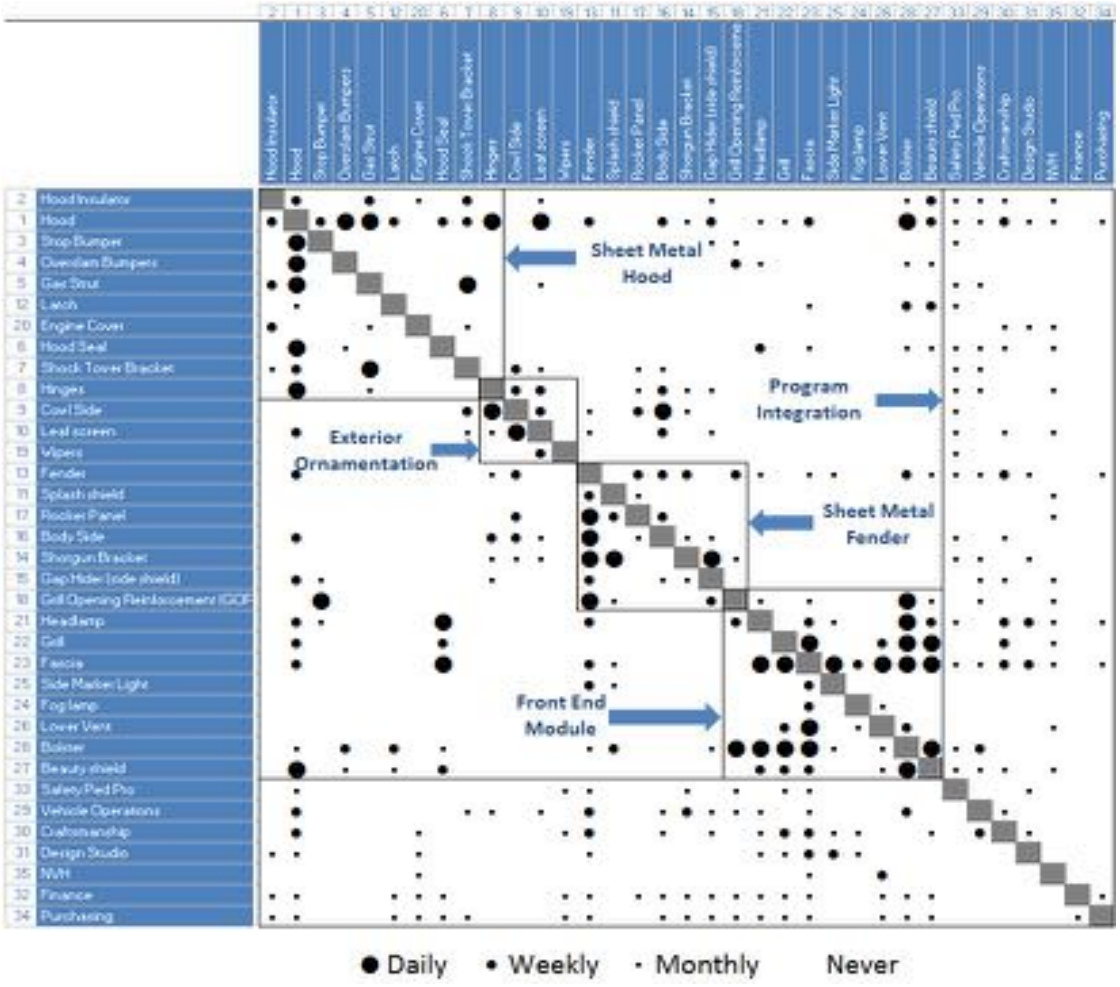




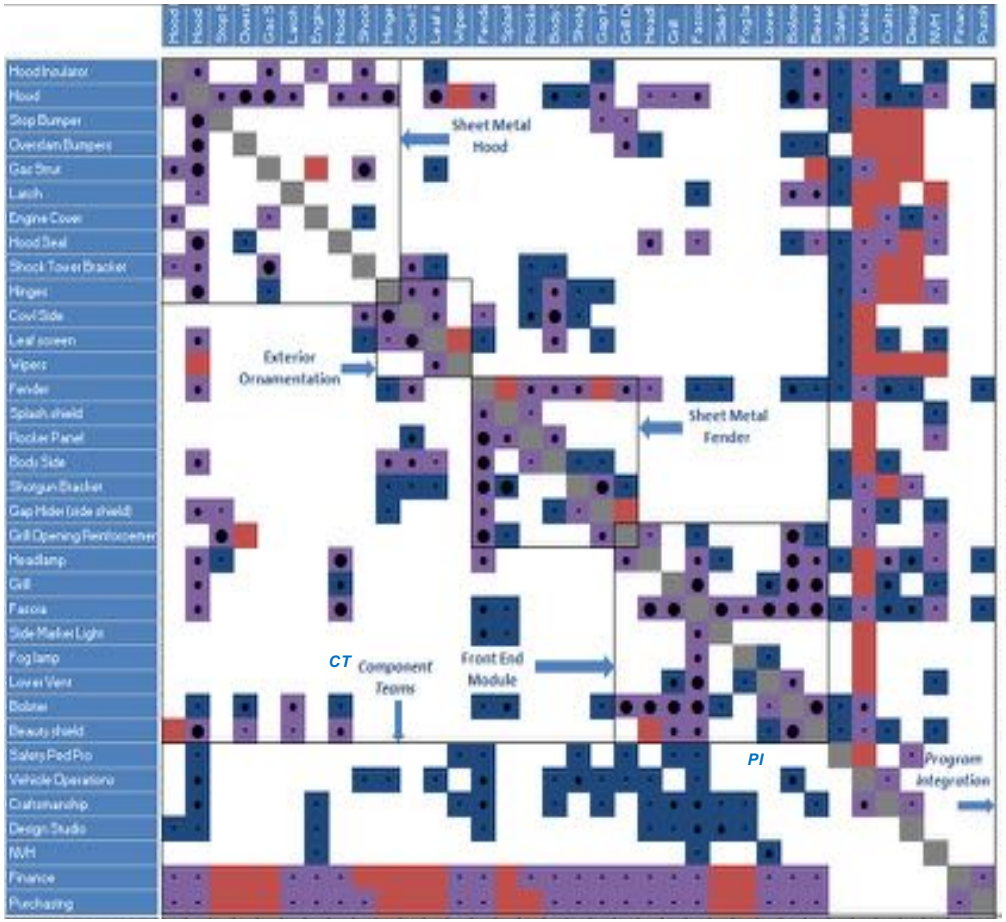
Product Architecture DSM

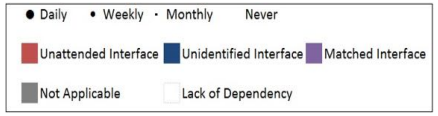


Organization Architecture DSM



Arch/Org Alignment DSM



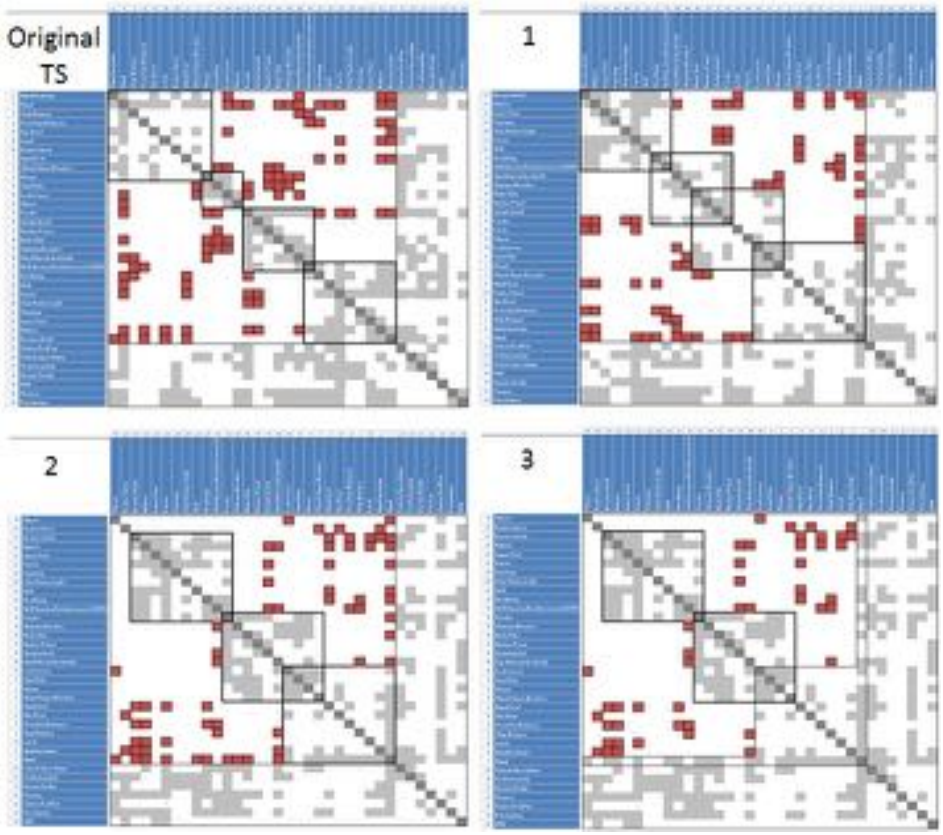


56% of team interactions matched by design interfaces

Team Communication (Organization Structure)	Component Interfaces (Product Architecture)			
	CT	PI	CT	PI
NO (887)	11	53	235	588
	Unattended (64)		Lack of Dependency (823)	
YES (338)	122	67	63	86
	Matched (189)		Unidentified (149)	
	YES (253)		NO (972)	

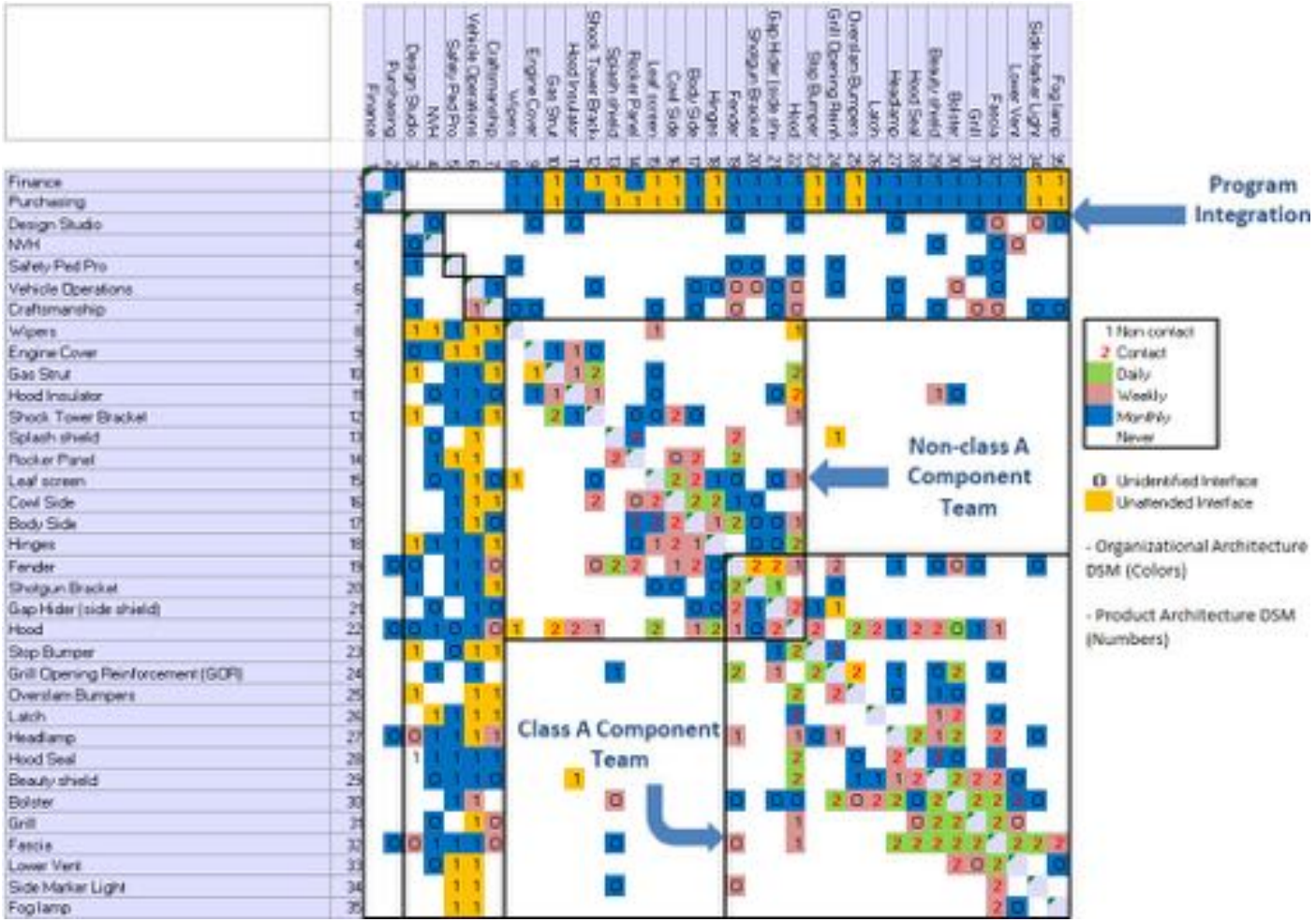
75% of design interfaces matched by team interactions

Simultaneous Arch/Org DSM Clustering



	CT	PI	Interactions outside	% Δ
Original TS	4	1	94	-
Proposal 1	4	1	72	23%
Proposal 2	3	1	65	31%
Proposal 3	3	2	46	51%
Final clustering	*2*	*1*	*8*	*91%*

Simultaneous Clustering Analysis



Technology Readiness Levels

Developed by NASA researcher Stan Sadin in 1974, TRLs:

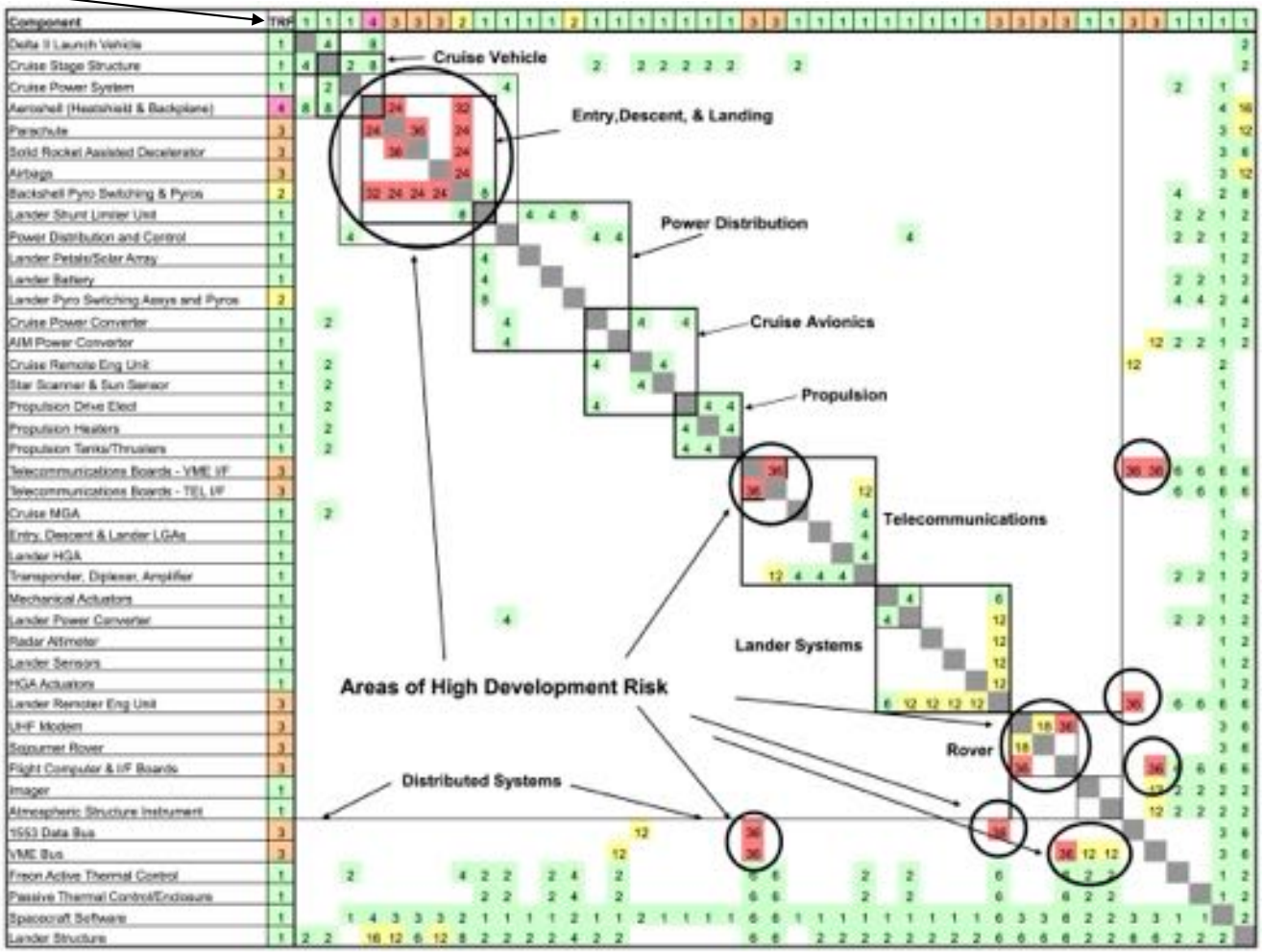
- **are technology discipline-independent metrics**
- **enable consistent comparison of maturity**
- **facilitate clear, well-documented assessments**
- **serve as basic guideposts for technology research and development**

TRL	Definition
9	Actual system “flight proven” through successful mission operations
8	Actual system completed and “flight qualified” through test and demonstration (ground or flight)
7	System prototype demonstration in a target/space environment
6	System/subsystem model or prototype demonstration in a relevant environment (ground or space)
5	Component and/or breadboard validation in relevant environment
4	Component and/or breadboard validation in laboratory environment
3	Analytical and experimental critical function and/or characteristic proof-of-concept
2	Technology concept and/or application formulated
1	Basic principles observed and reported

NASA Systems Engineering Handbook (NASA/SP-2007-6105 Rev 1)

Mars Pathfinder Technology Risk DSM

Component-level technology risk factor (TRF) based on each component's technology readiness level (TRL)



TR DSM value (i,j) = DSM(i,j) x TRF(i) x TRF(j)

Ref: Brady, TK, "Utilization of Dependency Structure Matrix Analysis to Assess Complex Project Designs," *ASME Design Engineering Technical Conferences*, Montreal, 2002 .

TRL DSM Exercise: System Readiness

Using the 1-9 TRL scale, how would you assess the "technology readiness" of each subsystem and of the system as a whole?

Report out:
4 SRL values.

TRLs

Subsystem	Component	#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Cruise and Launch	Delta II Launch Vehicle	1	9	X		X											
	Cruise Stage Structure	2	X	9	X	X											
	Cruise Power System	3		X	9							X					
Entry, Descent and Landing	Aeroshell (Heatshield & Backplane)	4	X	X		5	X			X							
	Parachute	5				X	6	X		X							
	Solid Rocket Assisted Decelerator	6					X	6		X							
	Airbags	7							6	X							
	Backshell Pyro Switching & Pyros	8				X	X	X	X	8	X						
Power Distribution	Lander Shunt Limiter Unit	9								X	9		X	X	X		
	Power Distribution and Control	10			X							9				X	X
	Lander Petals/ Solar Array	11									X		9				
	Lander Battery	12									X			9			
	Lander Pyro Switching & Pyros	13									X				8		
	Cruise Power Converter	14										X				9	
	AIM Power Converter	15										X					9

Failures Occur at Interfaces
– even with mature components –

European Space Agency Ariane 5 Rocket:
First Test Flight, June 4, 1996

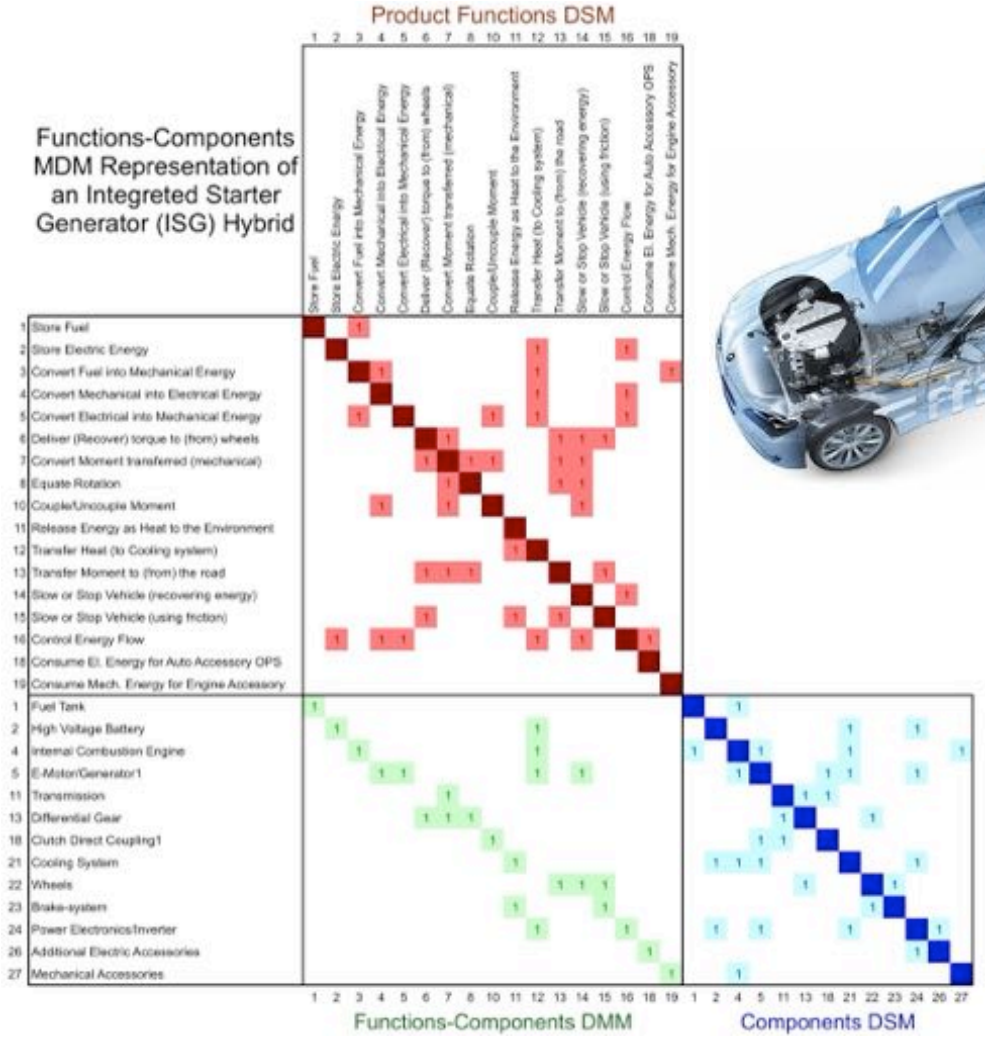
- Working code from the Ariane 4 rocket was reused in the Ariane 5, but the Ariane 5's faster engines triggered a bug in an arithmetic routine inside the rocket's flight computers.
- The error was in the code that converted a 64-bit floating-point number to a 16-bit signed integer. The faster engines caused the 64-bit numbers to be larger in the Ariane 5 than in the Ariane 4, triggering an overflow condition that resulted in the primary and backup flight computers crashing.
- As a result of two crashed computers, the rocket's primary processor overpowered the rocket's engines and caused the rocket to disintegrate 40 seconds after launch.

Ref: *Wired*, "History's Worst Software Bugs"

Multi-Domain DSM Models

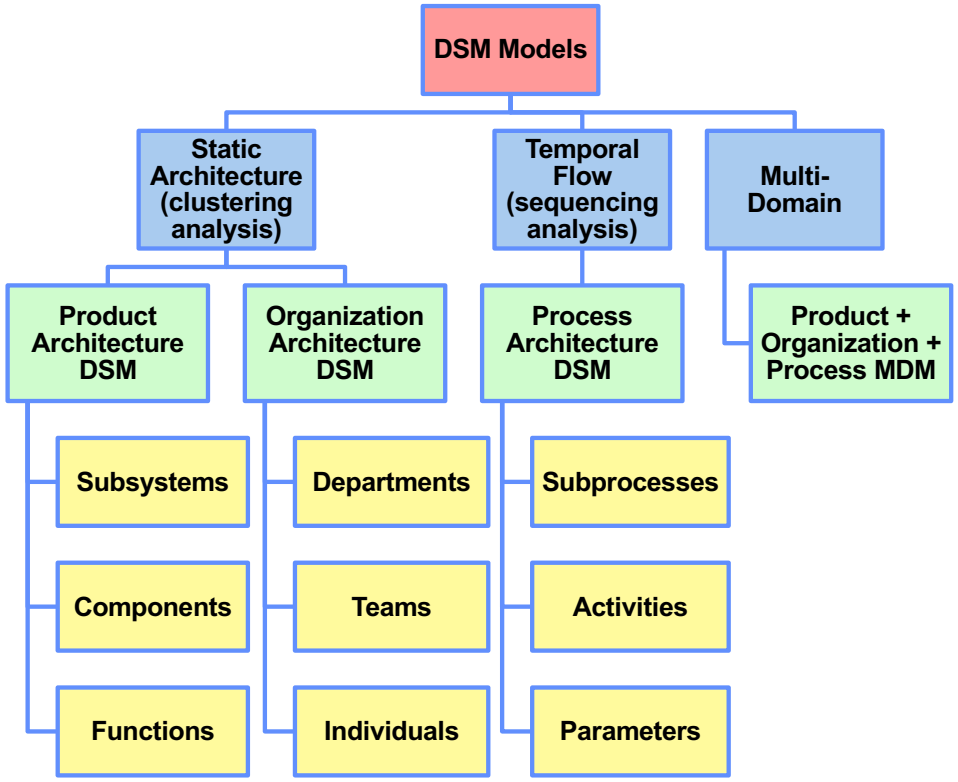
Goals DSM g x g	Goals-Product DMM g x d	Goals-Process DMM g x p	Goals-Organization DMM g x o	Goals-Tools DMM g x t
	Product DSM d x d	Product-Process DMM d x p	Product-Organization DMM d x o	Product-Tools DMM d x t
		Process DSM p x p	Process-Organization DMM p x o	Process-Tools DMM p x t
			Organization DSM o x o	Org.-Tools DMM o x t
				Tools DSM t x t

BMW Hybrid Engine Starter-Generator MDM Model



Ref: Gorbea, et al., "Analysis of Hybrid Vehicle Architectures using Multiple Domain Matrices," *International DSM Conference*, Stockholm, 2008.

Types of DSM Models and Analysis

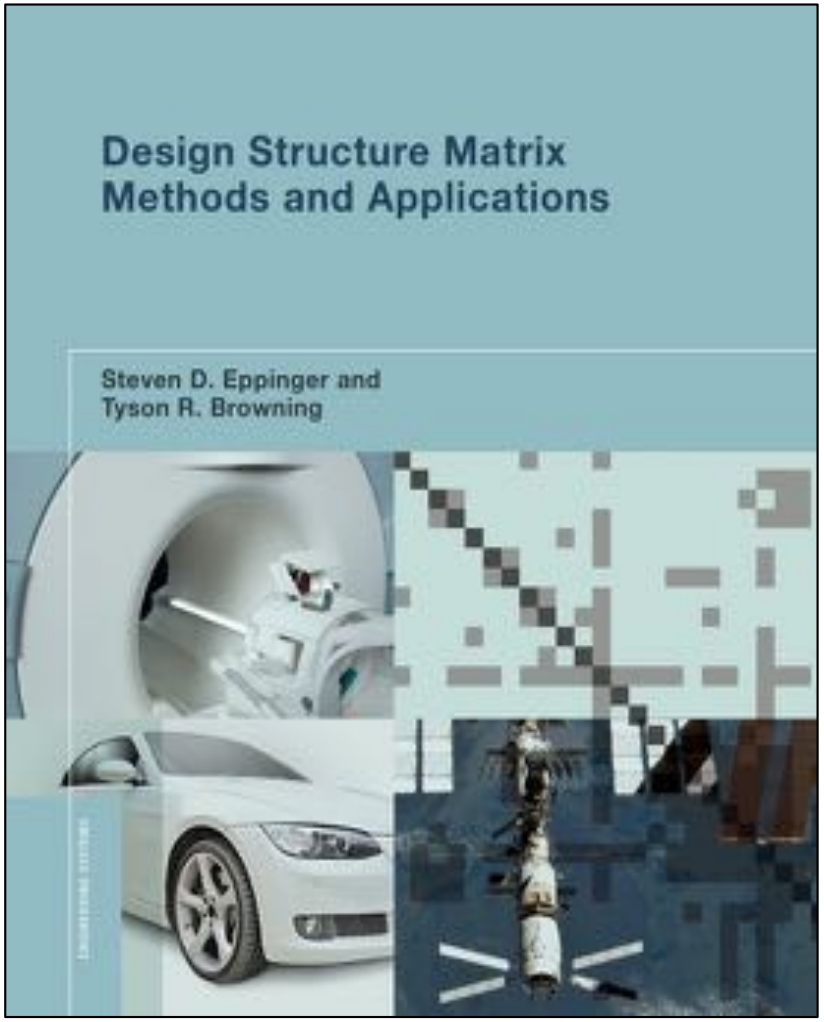


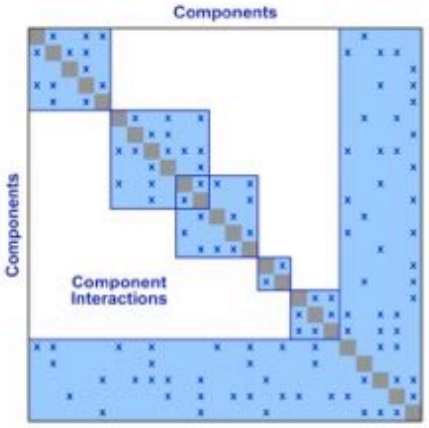
How to Learn More about DSM

- DSMweb.org Online Community
- Software Tools
- New Research
- Workshops and Consulting
- Annual DSM Conference
- Reference Publications

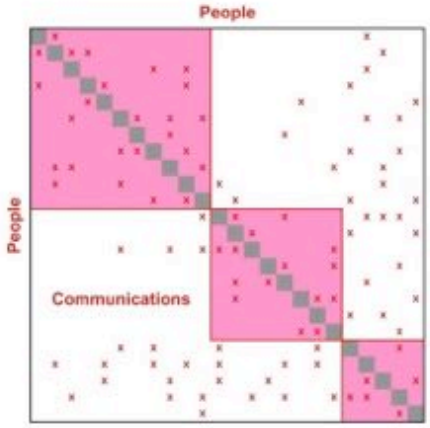
DSMweb.org


The Design Structure Matrix (DSM)
HOME
ABOUT DSM
UNDERSTAND DSM
DSM KNOWLEDGE
DSM TOOLS
DSM COMMUNITY
CONTACT
Technical DSM Tutorial
Contact
Community
Software
Publications
Tutorial
Home

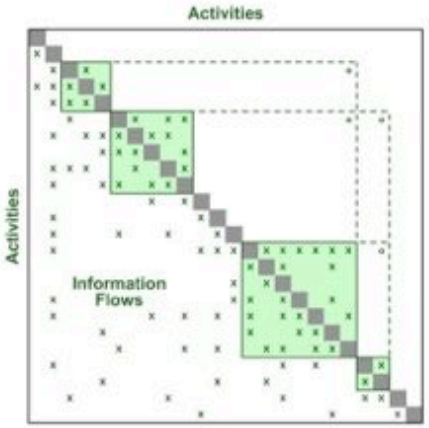




Product Architecture DSM



Organization Architecture DSM



Process Architecture DSM

Multi-Domain MDM

Annual DSM Conference

dsm-conference.org

Oct. 2022

MIT

UNIVERSITY OF TRIESTE

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

NPS NAVAL POSTGRADUATE SCHOOL

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

TCU

A! Aalto University

TECHNISCHE UNIVERSITÄT MÜNCHEN

CLEMSON UNIVERSITY

KUNGL TEKNISKA HÖGSKOLAN

UNIVERSITY OF CAMBRIDGE

BOEING

USP

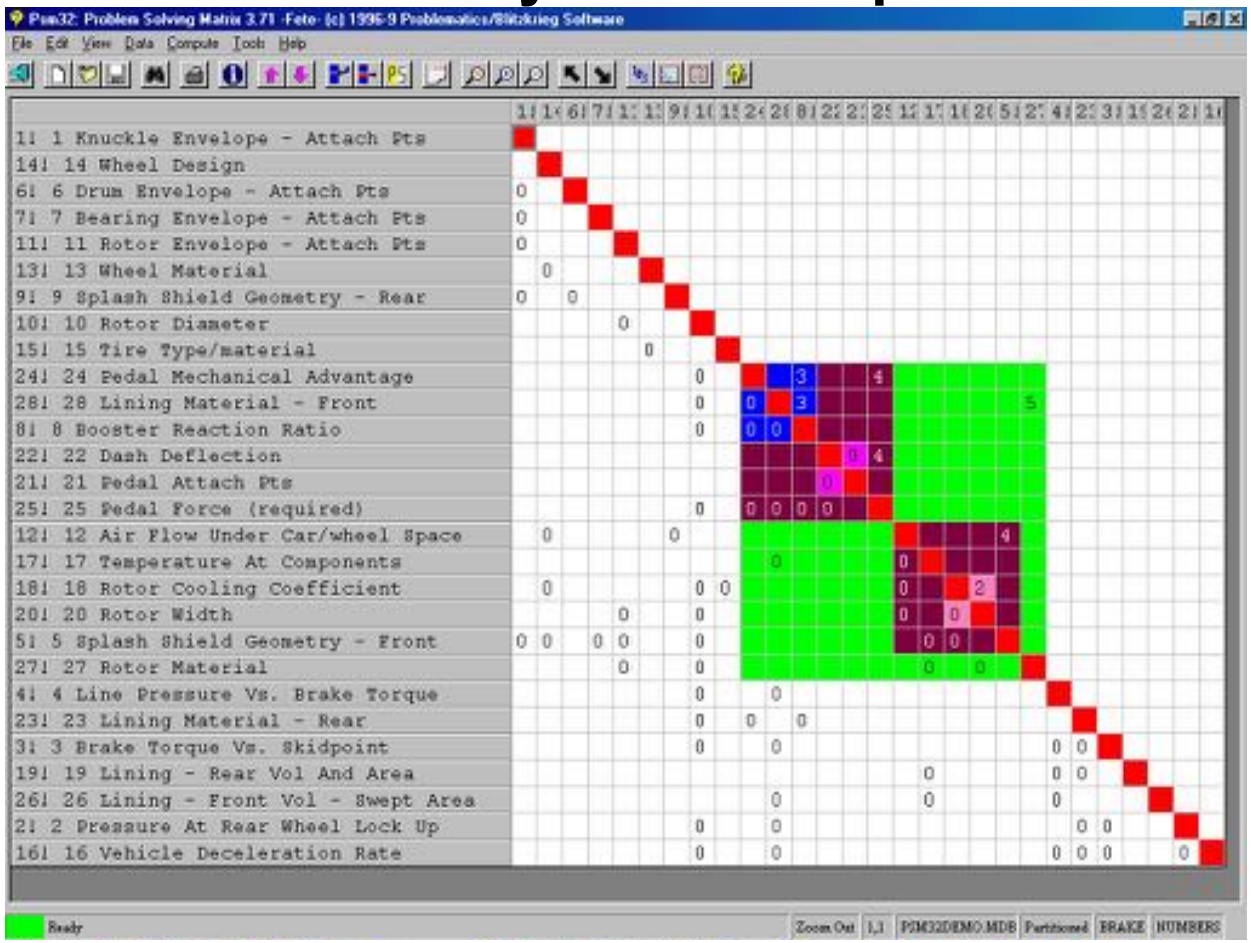
iSiD

CENTRALE PARIS

the Design Society
a worldwide community

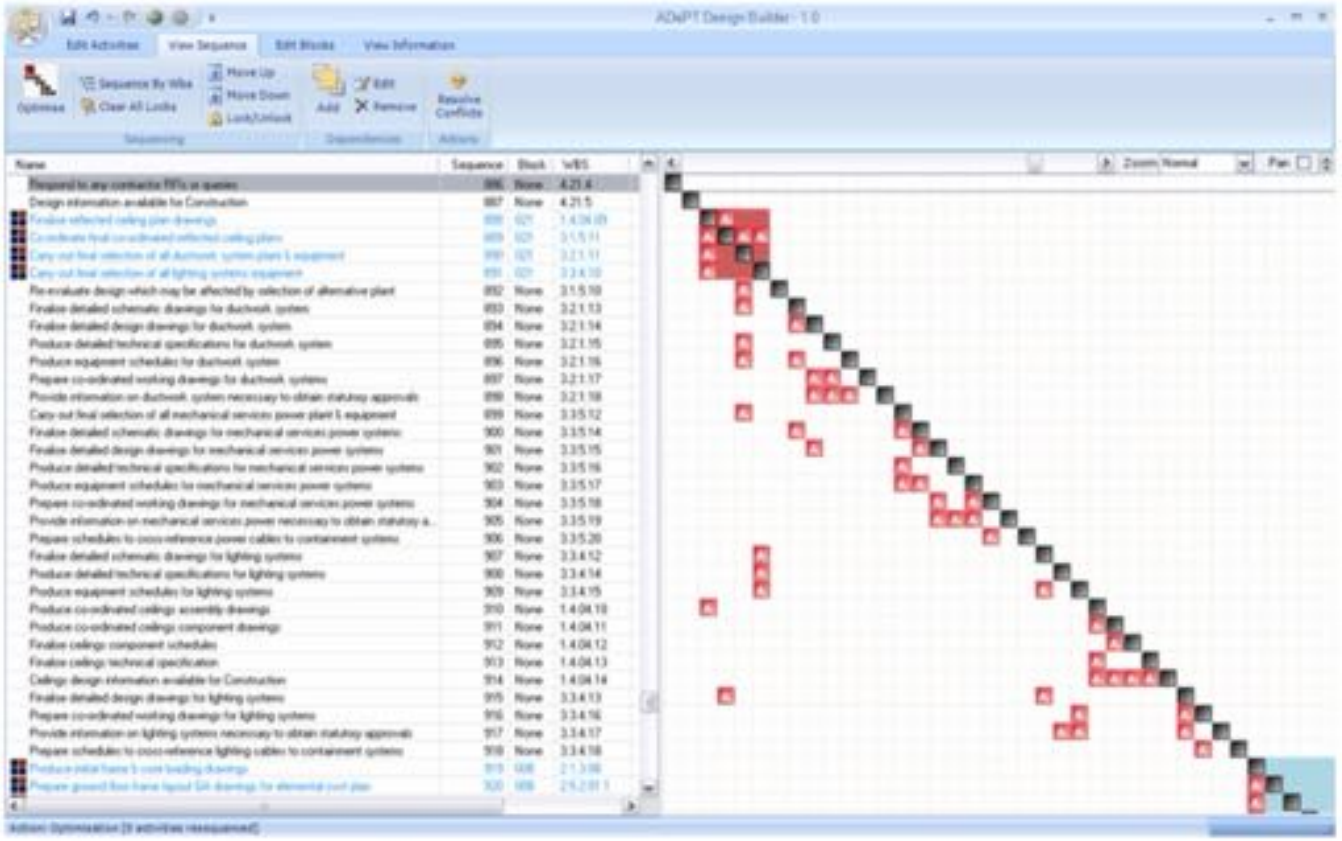
Project Management Institute.

PSM32 DSM Analysis Software: Brake System Example



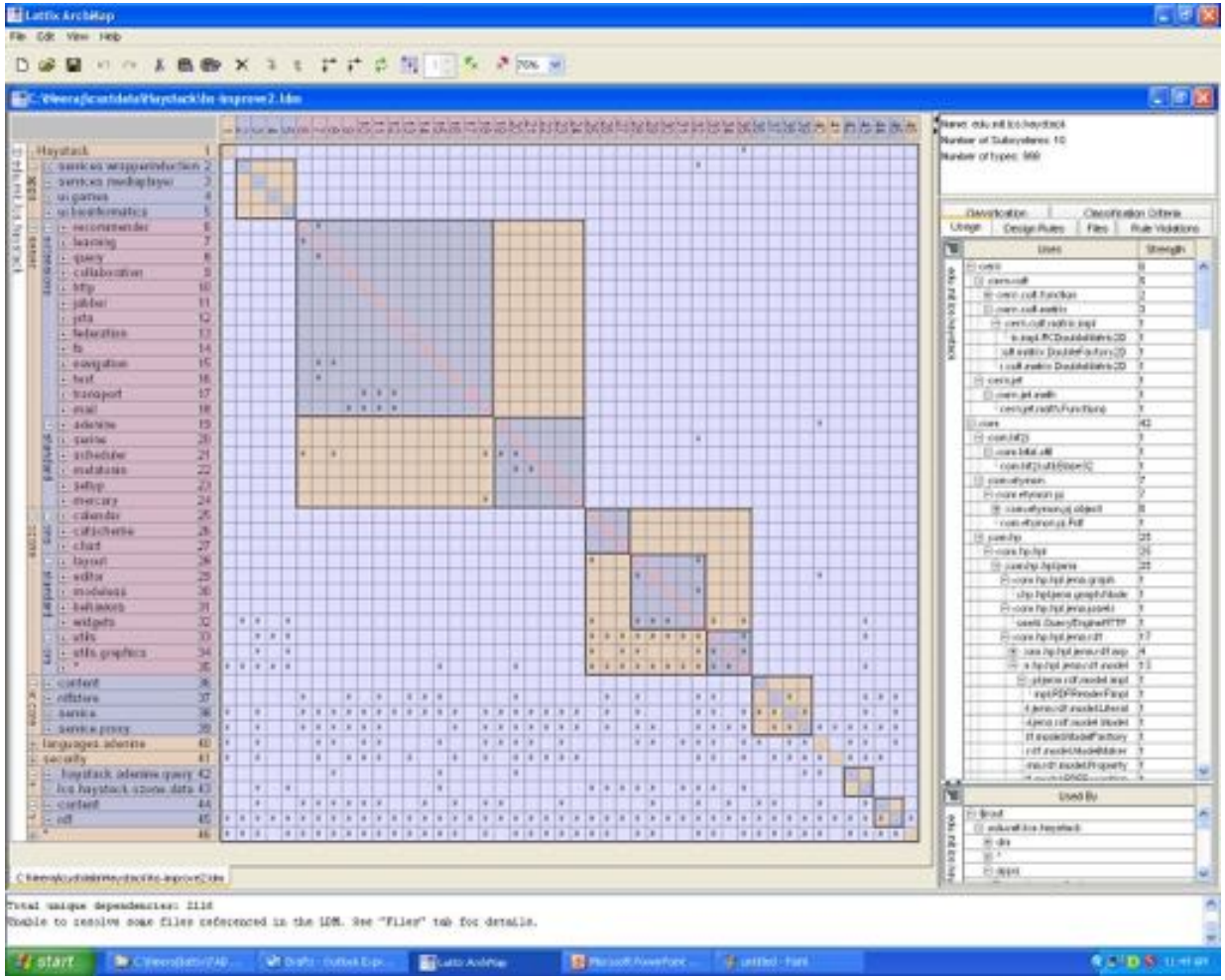
problematics.com

ADePT DSM Software



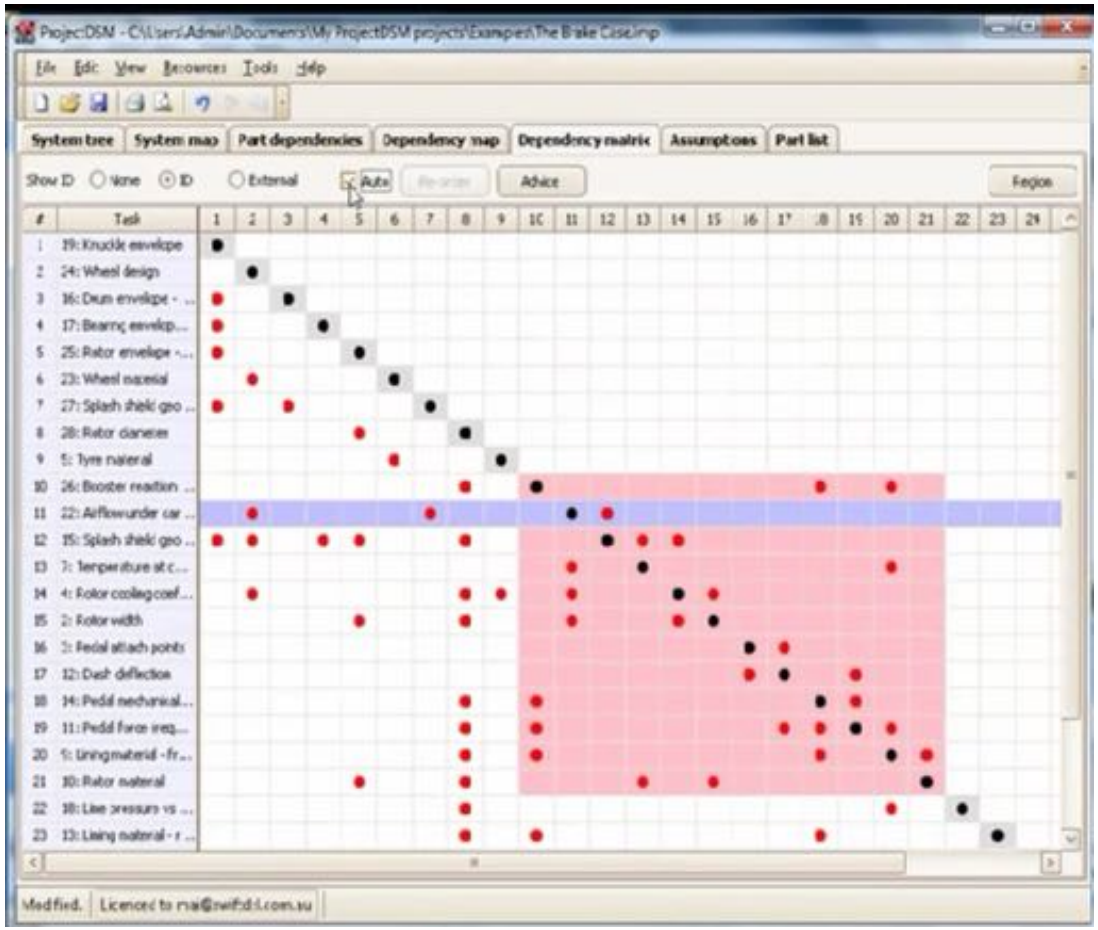
software.adeptmanagement.org

Lattix DSM Software



LATTIX

ProjectDSM Software



Note:
Free, demo, and pro versions available at projectdsm.com

contact: Mark Irving < mark.irving@projectdsm.com>

projectdsm
Design▪Structure▪Manage

Discussion

- What are potential applications for DSM in your organization?
- How could you begin?
- What do you hope to achieve?

Managing Complex Technical Projects

Day 3: Agile Development Methods

Prof. Steven D. Eppinger

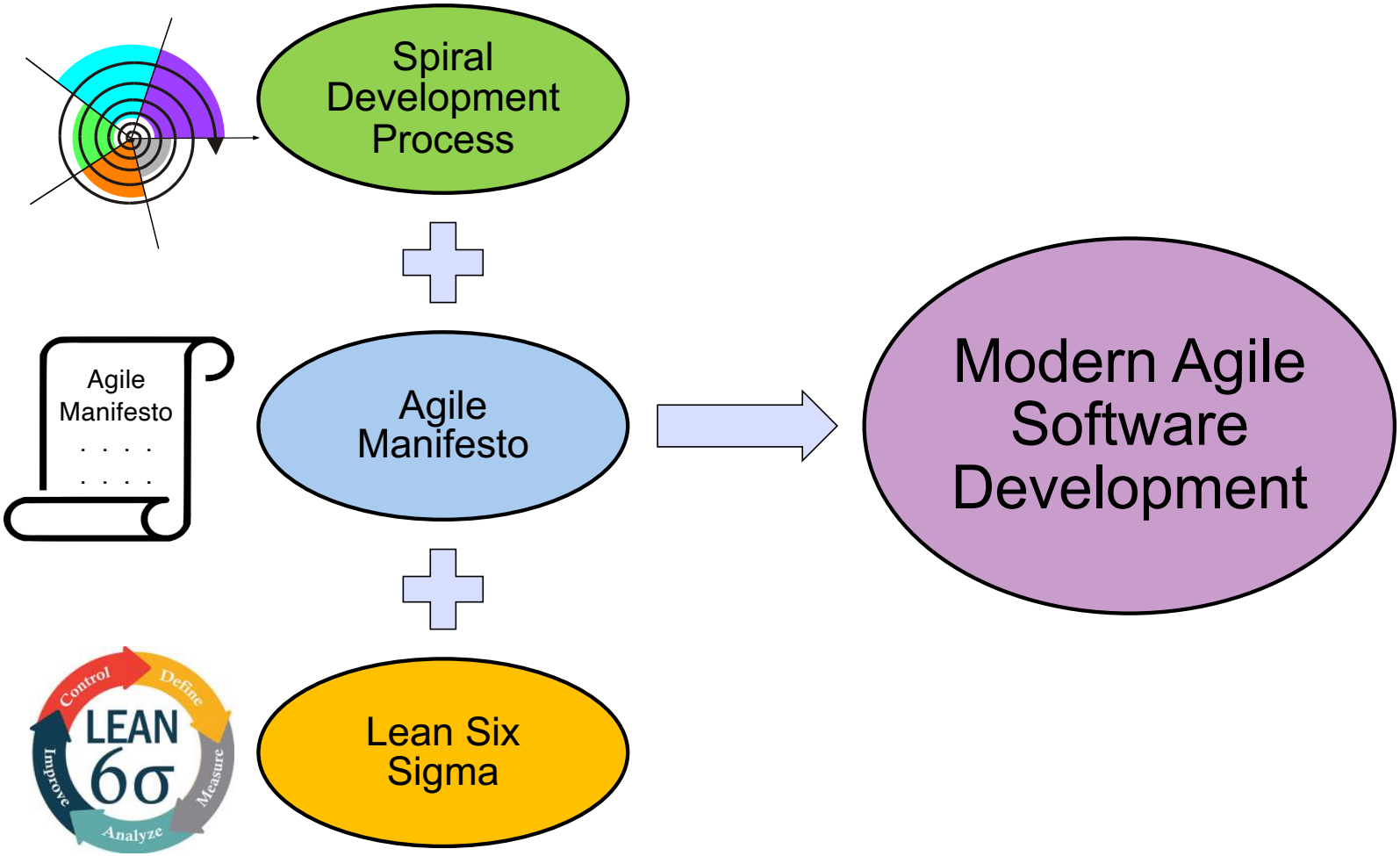
Massachusetts Institute of Technology

Sloan School of Management

eppinger@mit.edu

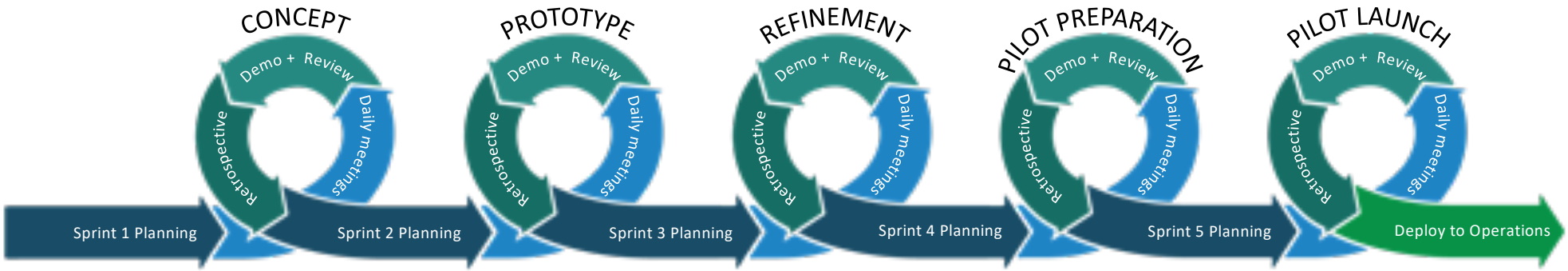
web.mit.edu/eppinger

www.dsmweb.org



Three *Roots* of Agile Software Development

Snack Food Development: Time-Boxed Sprints



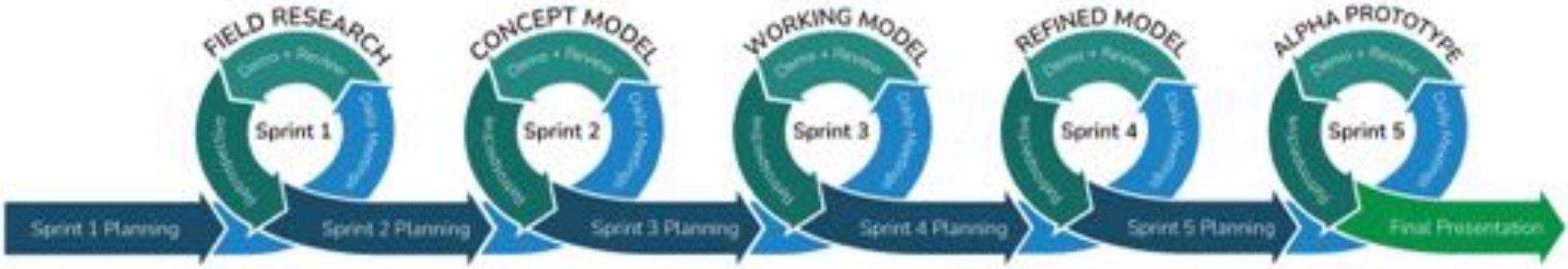
PDD2022
Product Design & Development

MIT Product Design and Development Class Projects: Time-Boxed Sprints

Smart Mirror

AR for Bicyclists

Infant Monitor



10 Agile Ideas You Can Use

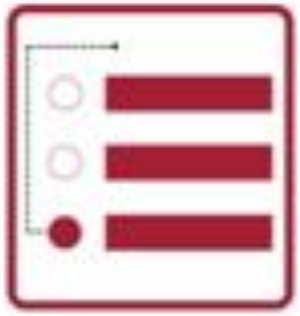
Time-Boxed Sprint

Sprint Planning

Daily Meeting

Sprint Demo/Review

Sprint Retrospective



Product Backlog

Sprint Backlog

Scrum Team

Team Leadership

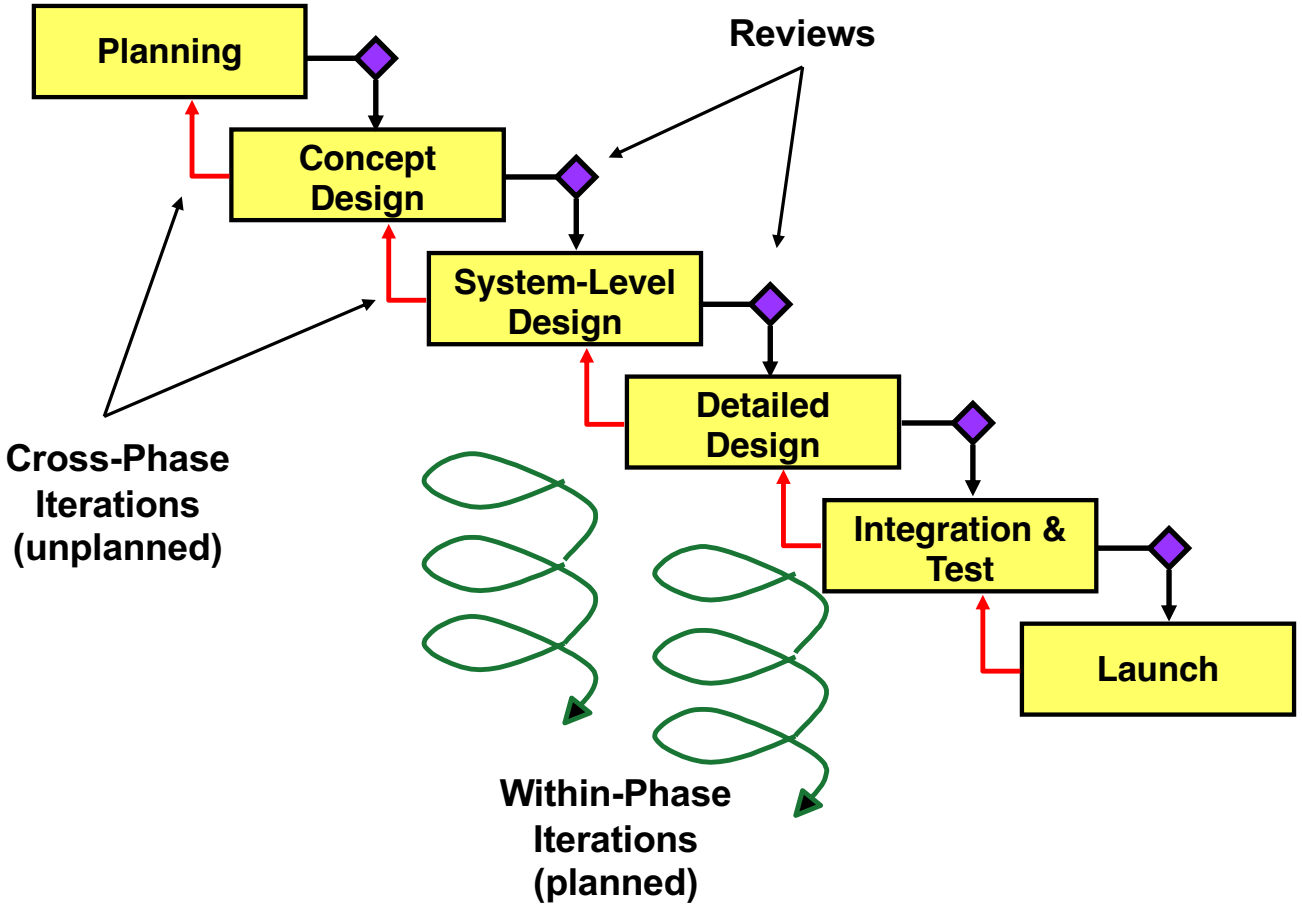
Agile Values

The Agile Manifesto:
A statement of software project values

We value	Individuals and interactions	over	Process and tools
We value	Working software	over	Comprehensive documentation
We value	Customer collaboration	over	Contract negotiation
We value	Responding to change	over	Following a plan

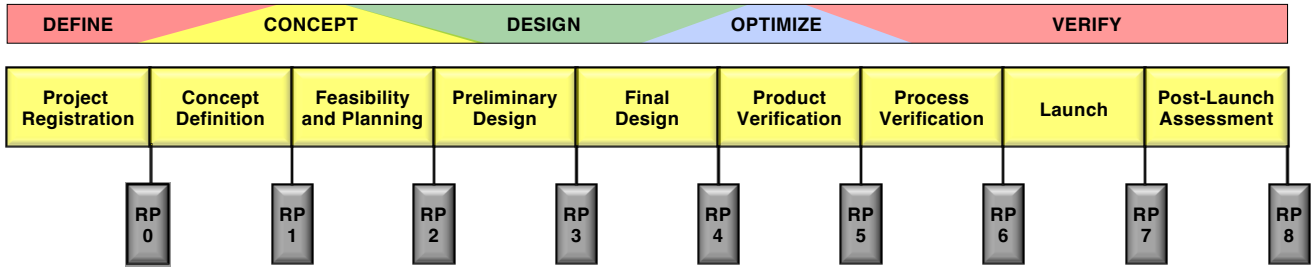
www.agilemanifesto.org

Staged Development Process



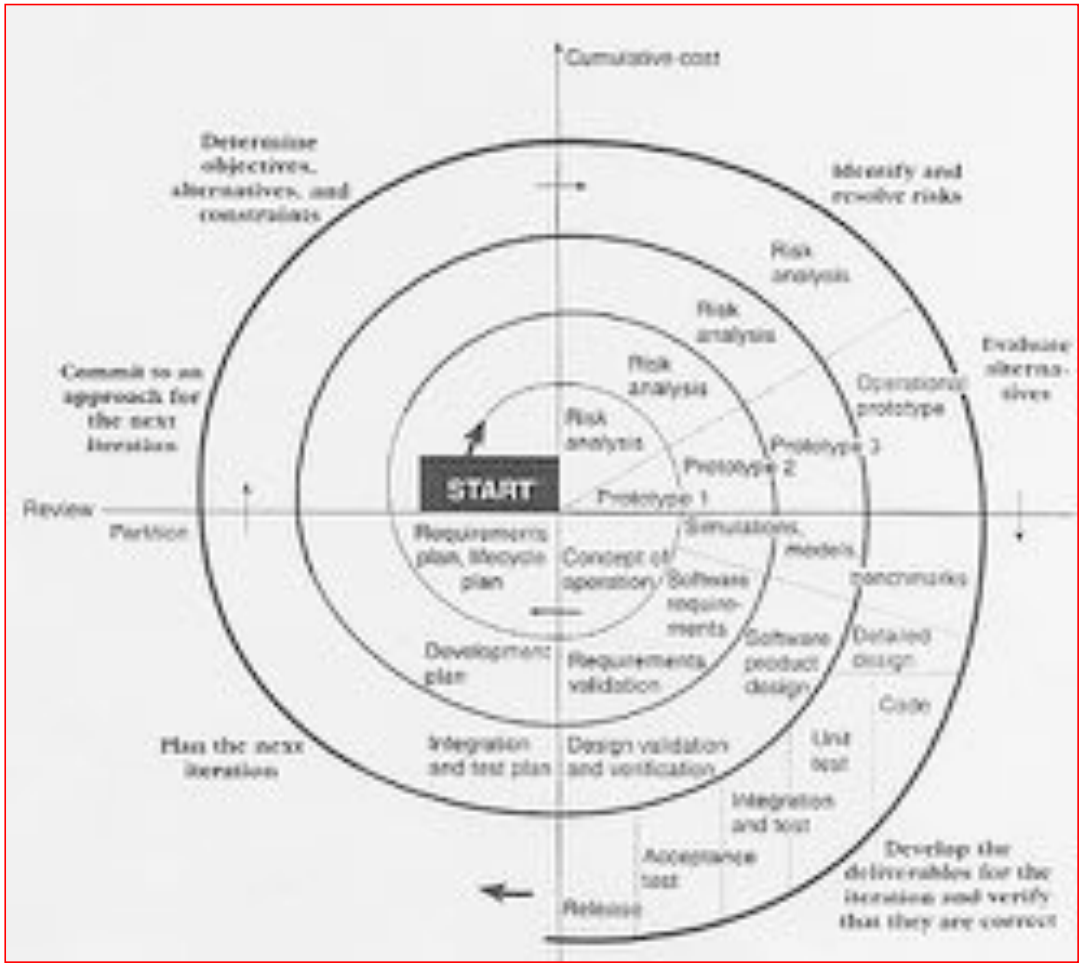
Ref: Robert Cooper, Winning at New Products 3rd ed., 2001.

Tyco International *Rally Point* PD Process



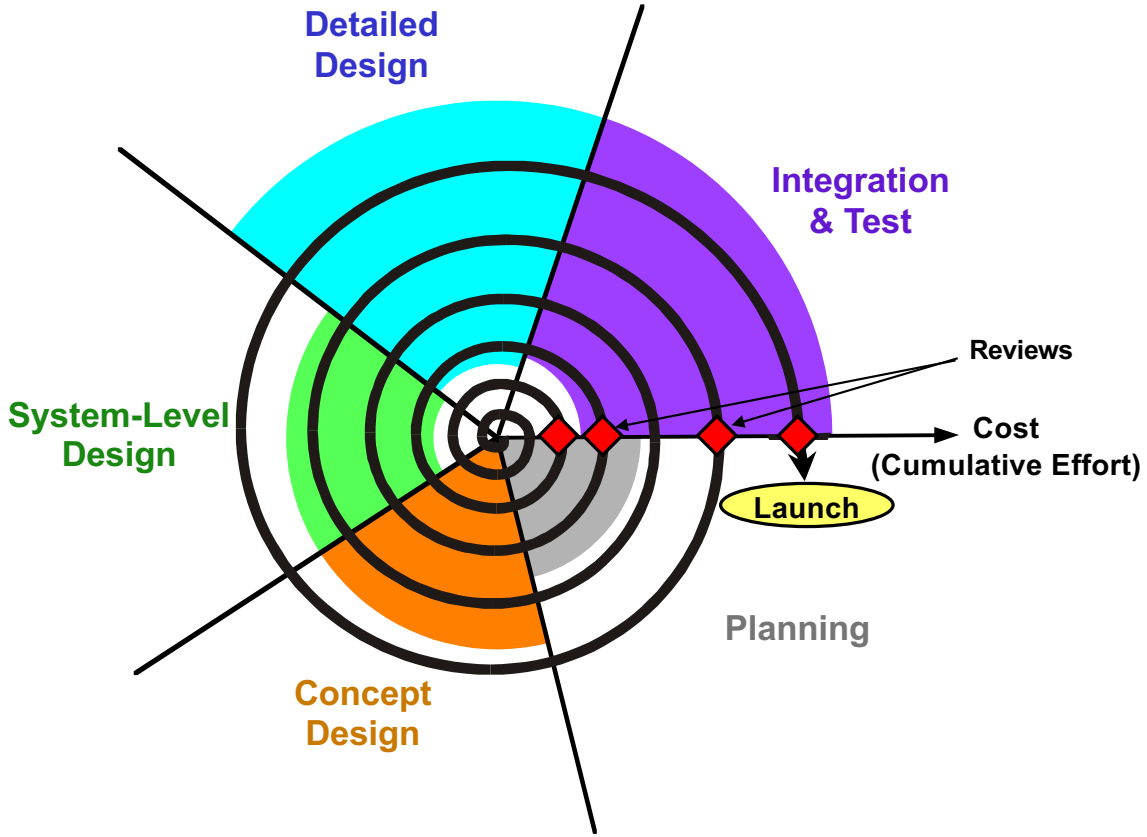
tyco

Spiral Development Process



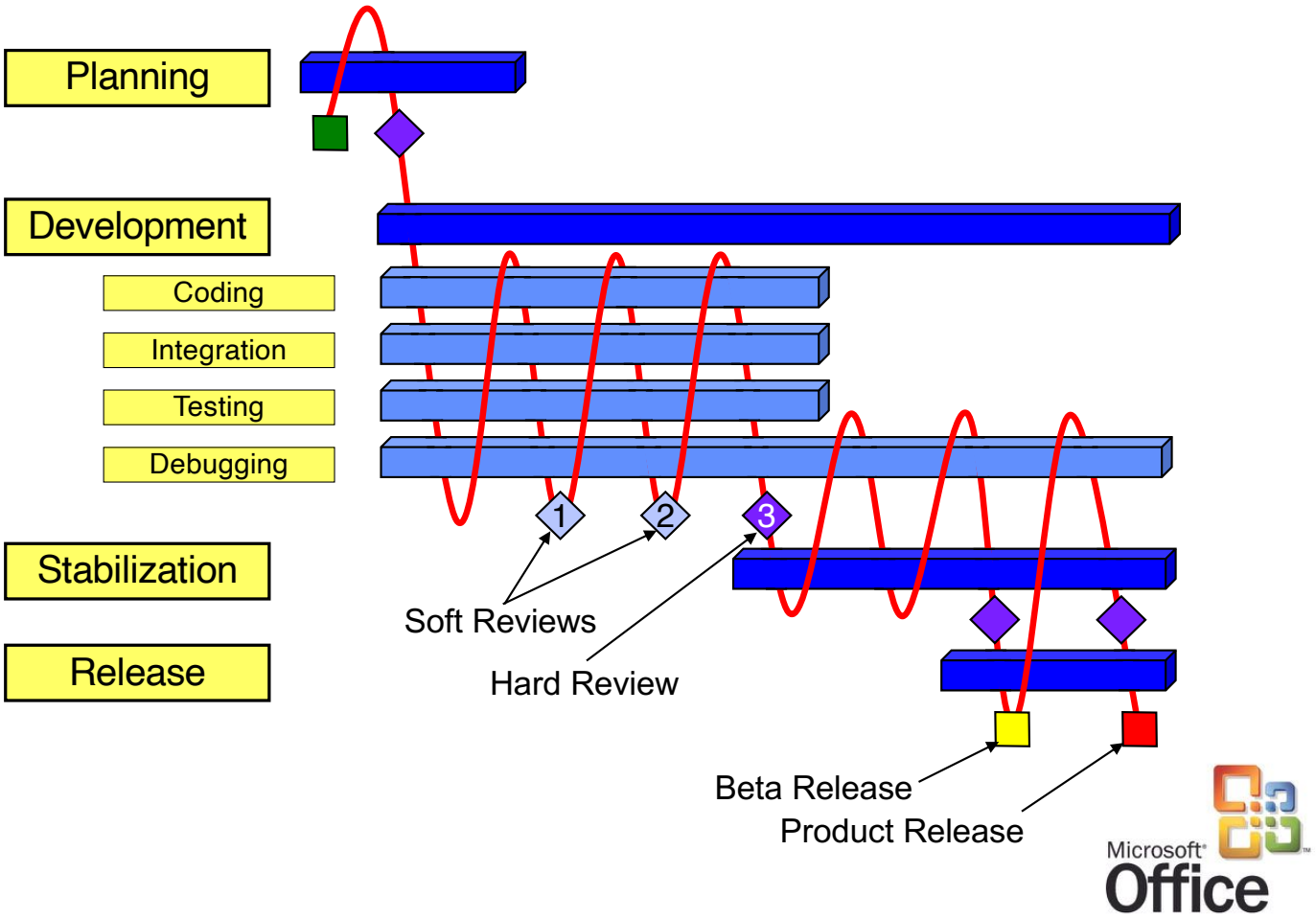
Ref: Barry Boehm, *A Spiral Model of Software Development and Enhancement*, 1988.

Spiral Development Process

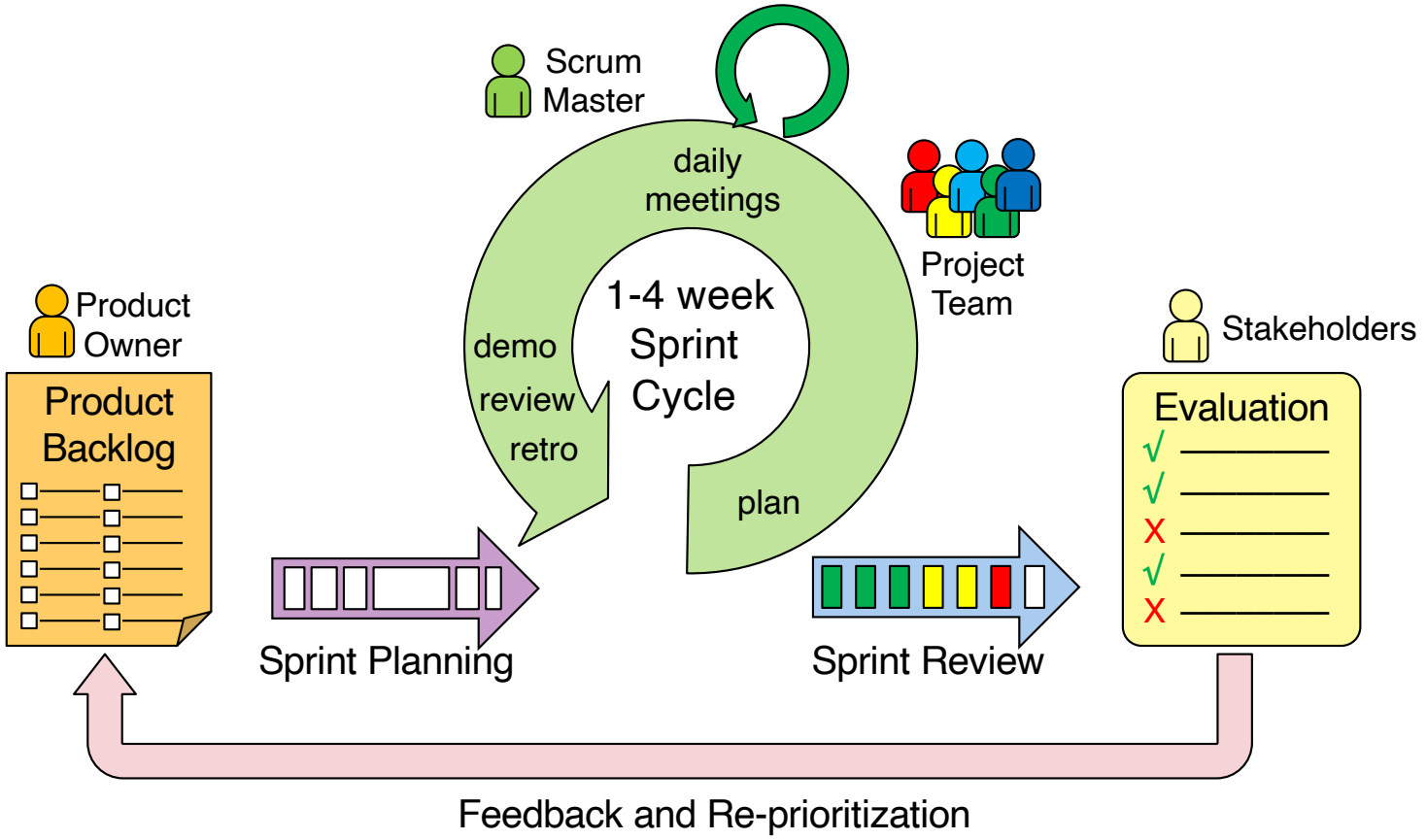


Ref: Barry Boehm, *A Spiral Model of Software Development and Enhancement*, 1988.

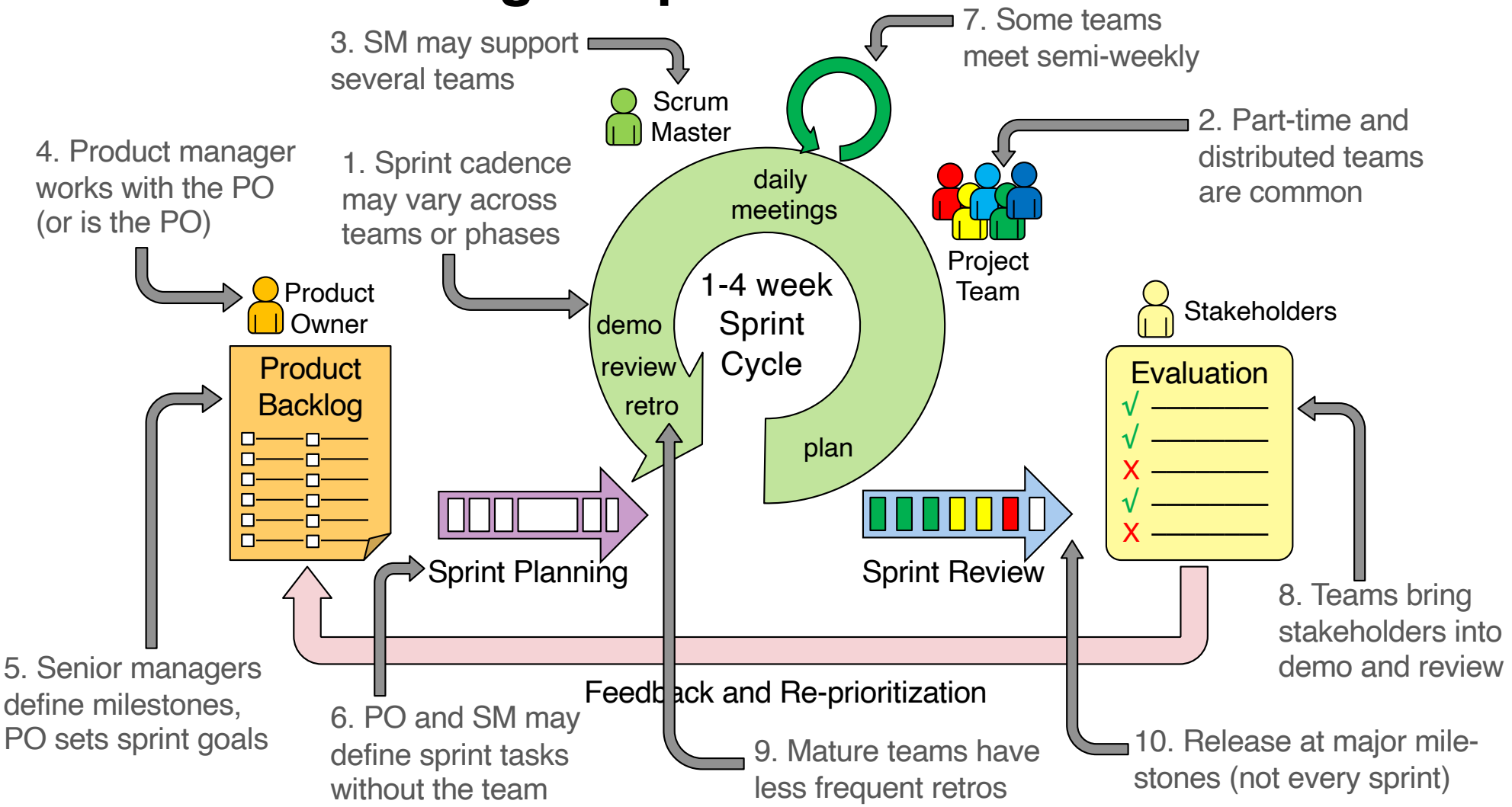
Microsoft *Milestone Build* Spiral Process



Agile Sprint Process



Adjustments to the **Agile Sprint Process**



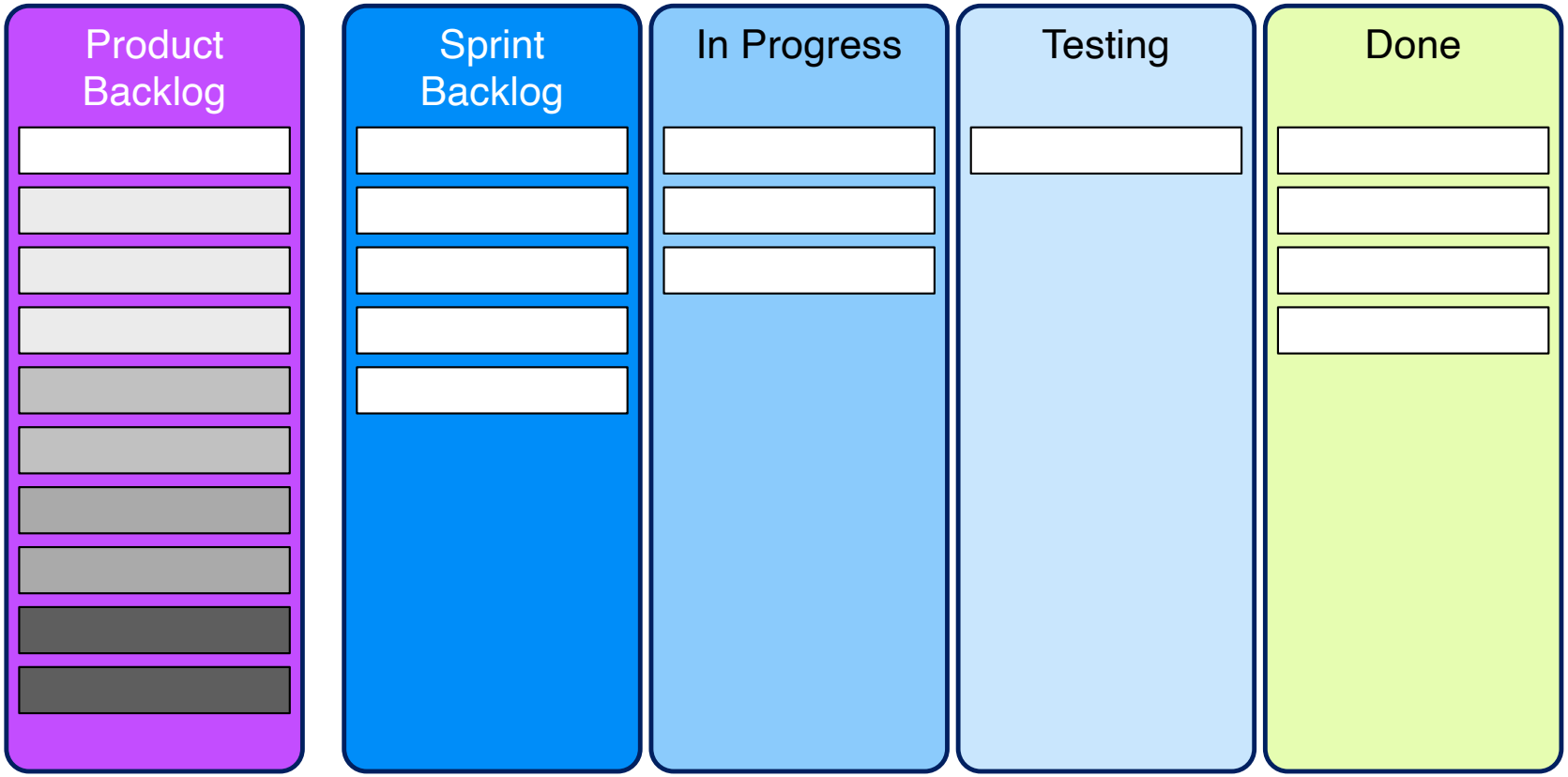
Demonstration of a Daily Meeting

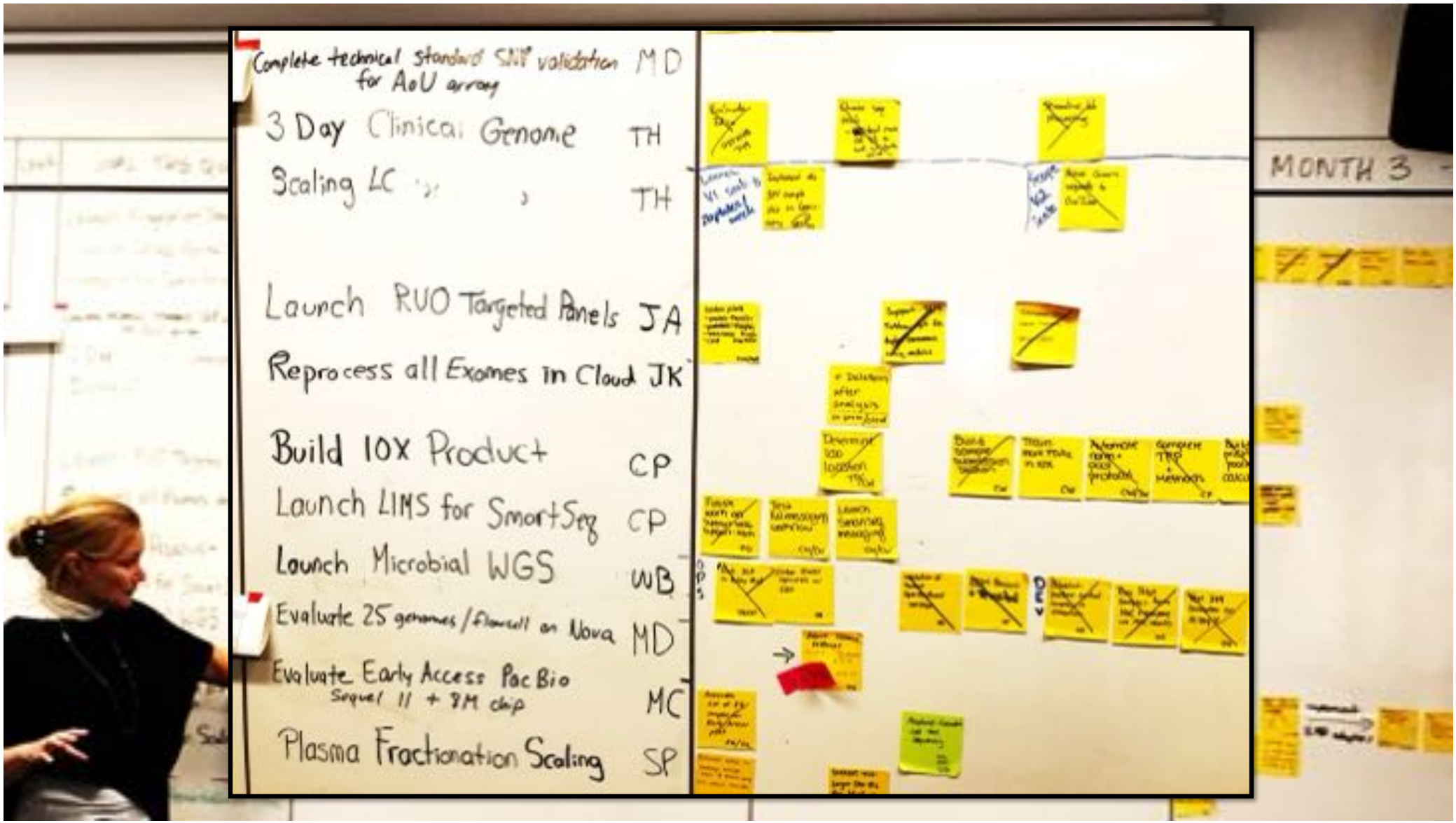
How a PD team uses the daily meeting to:

- Focus on value to customers
- Stay informed of status and actions
- Understand project risks
- Keep it short

Visual Management:

Kanban or program board makes status clear




Complete technical standard SNP validation for AoU array MD
3 Day Clinical Genome TH
Scaling LC TH
Launch RUO Targeted Panels JA
Reprocess all Exomes in Cloud JK
Build 10X Product CP
Launch LIMS for SmartSeq CP
Launch Microbial WGS WB
Evaluate 25 genomes/flowcell on Nova MD
Evaluate Early Access PacBio Sequel II + 8M chip MC
Plasma Fractionation Scaling SP
MONTH 3

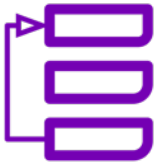

PDD 2022 Staff Board
Future Backlog
Soon Backlog
Current Backlog
In Progress
Week -1 Done
Week 1 Done
Week 2 Done
Week 3 Done
January Done
PDD2022

Backlog Planning

Four Principles of Backlog Planning

1. Short-term goals
   - Major milestones (releases) flow down to sprint goals

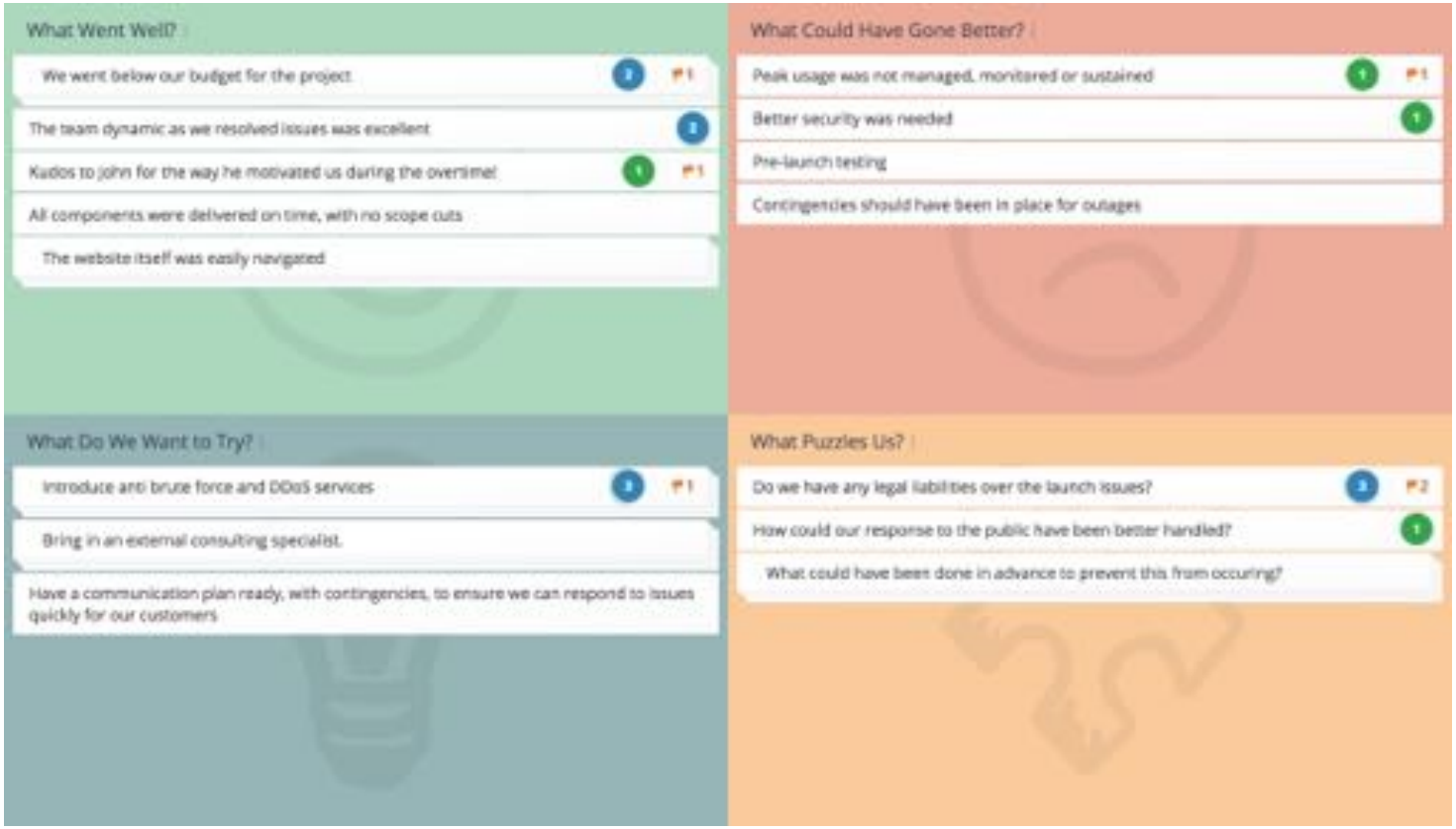
2. Decomposition
   - Goals break down into tasks, jobs, or stories



3. Prioritization
   - Product owners re-prioritize the backlog prior to sprint planning

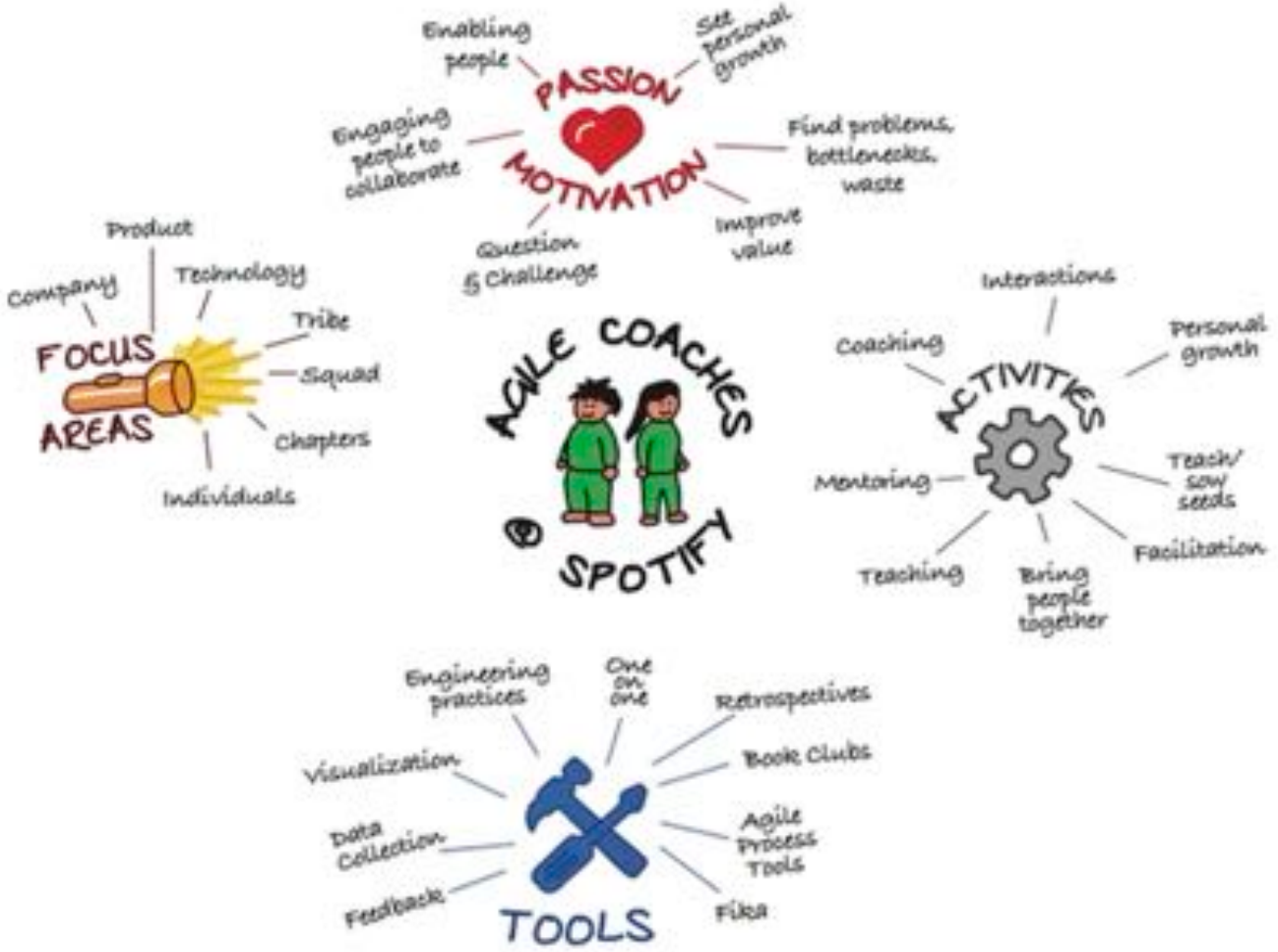
4. Refinement
   - Team defines each task (story) size, scope, and acceptance criteria (definition of done)

Sprint Retrospective

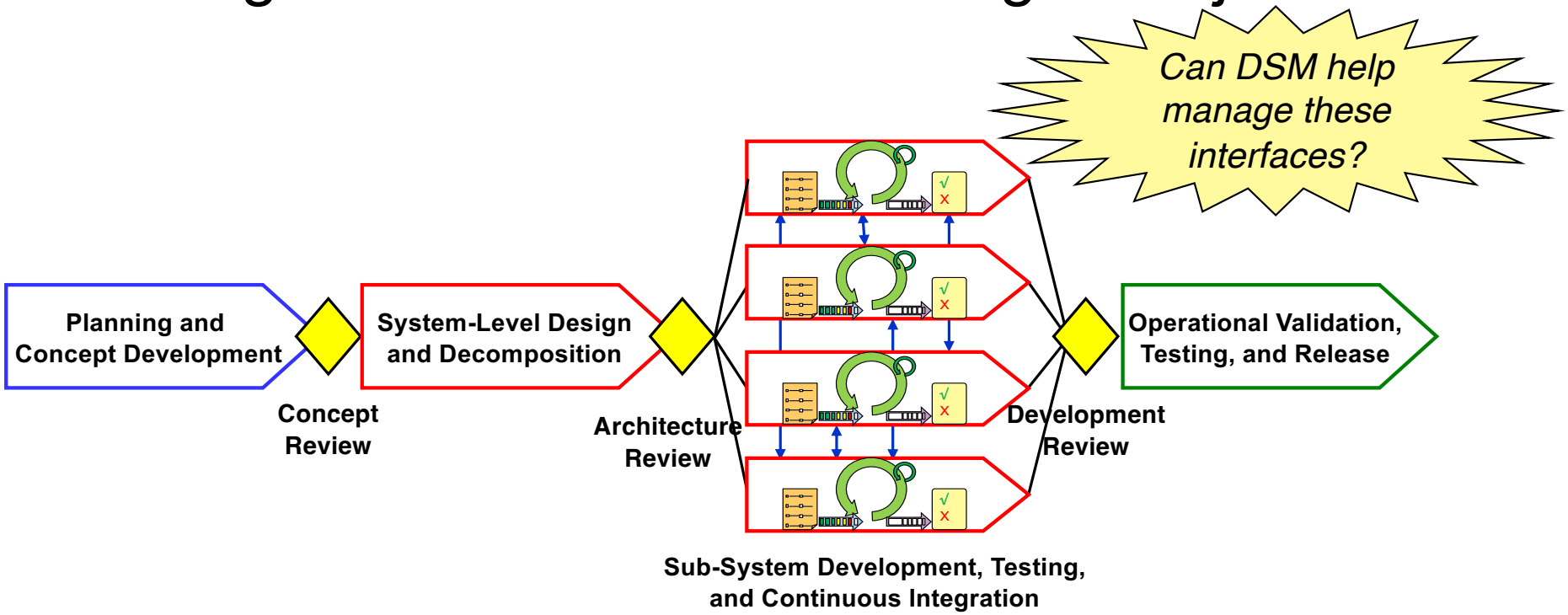


The final event of every sprint is a brief retrospective meeting to discuss ideas for continuous improvement of the process – ideally implemented in the next sprint.

Agile Coaching

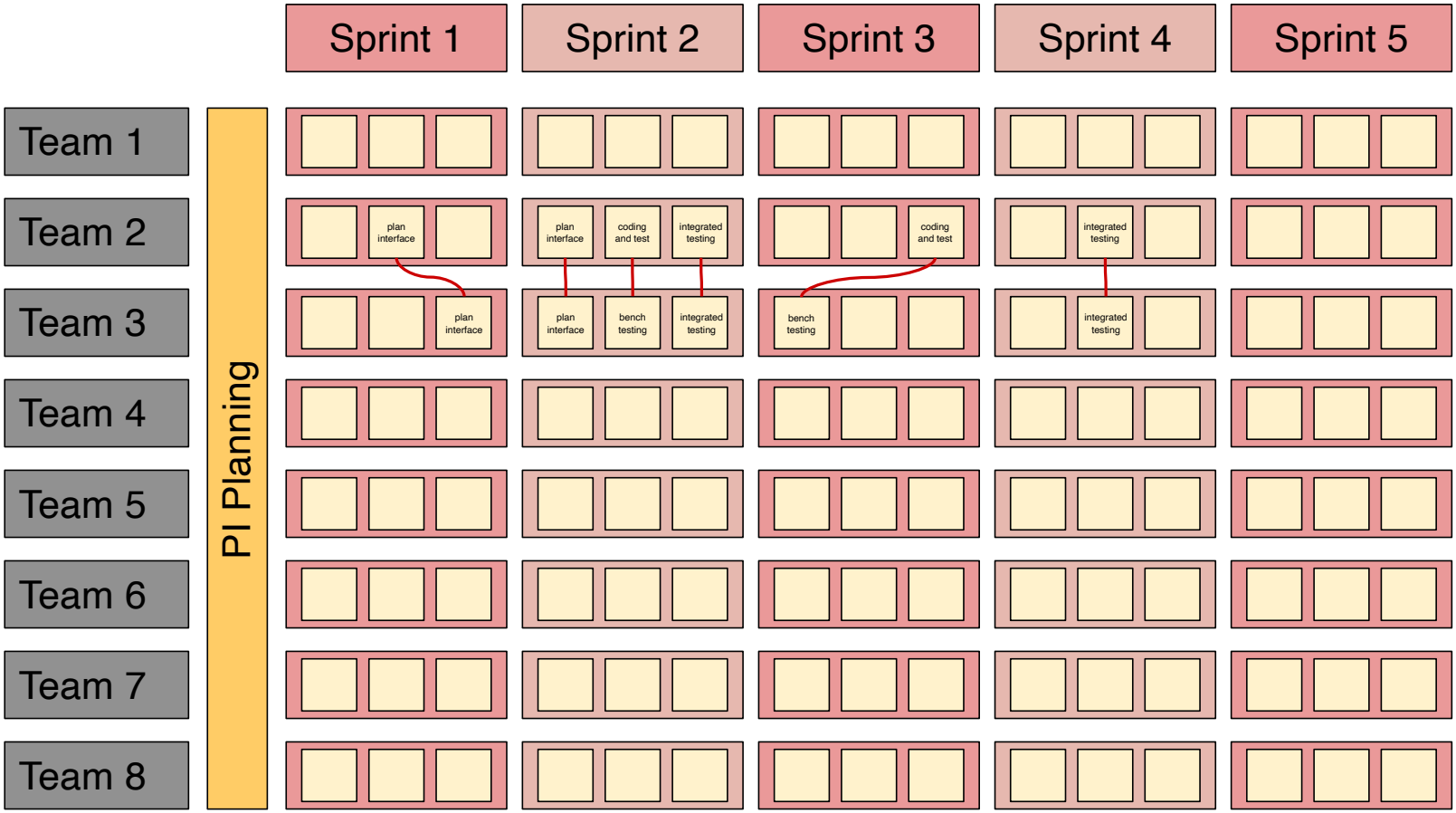


Scaled Agile:
Planning and Coordination for Larger Projects

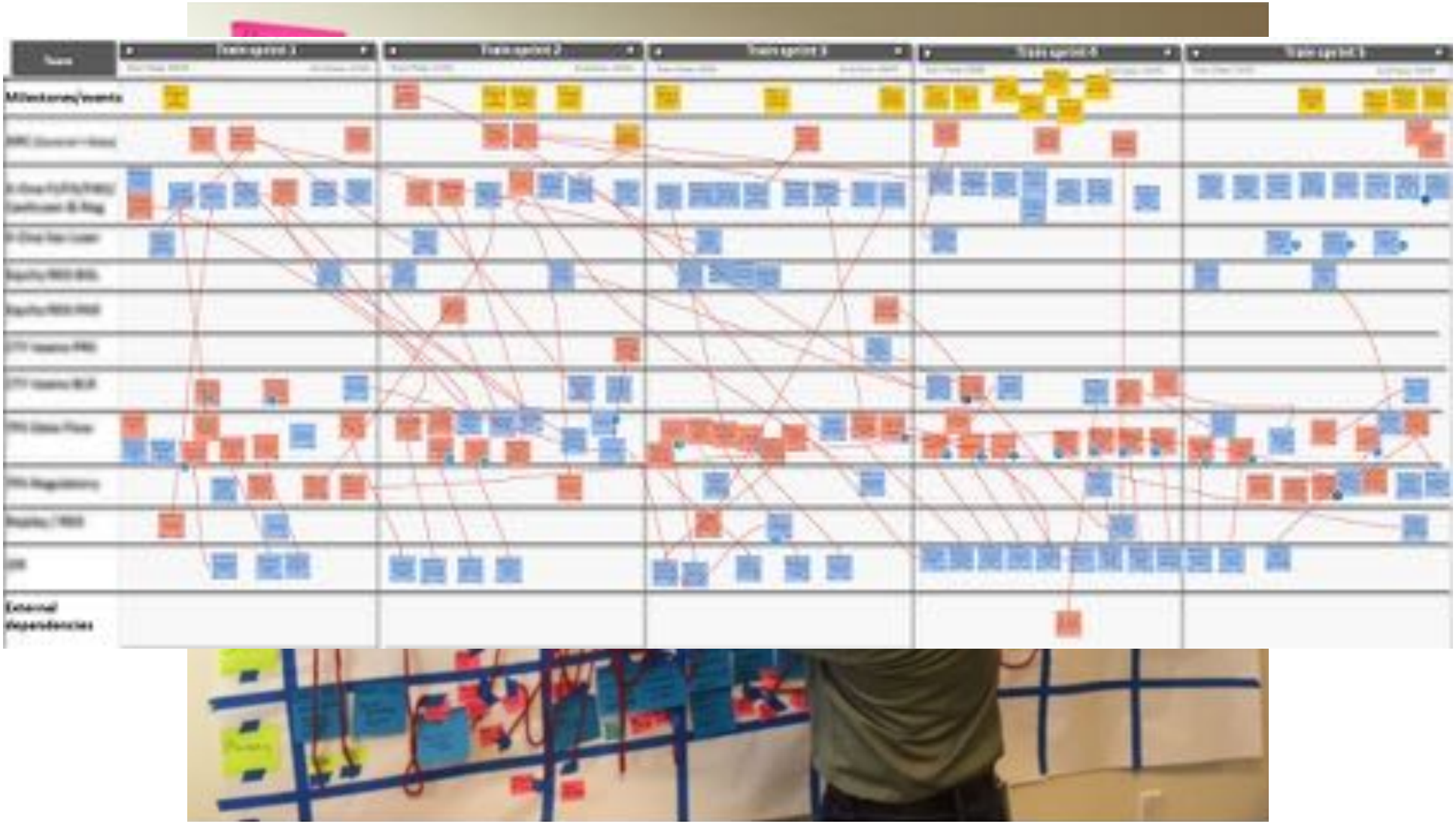


- Multi-team, multi-sprint planning helps to identify interfaces across teams.
- Scrum Masters and Product Owners handle emergent interfaces.

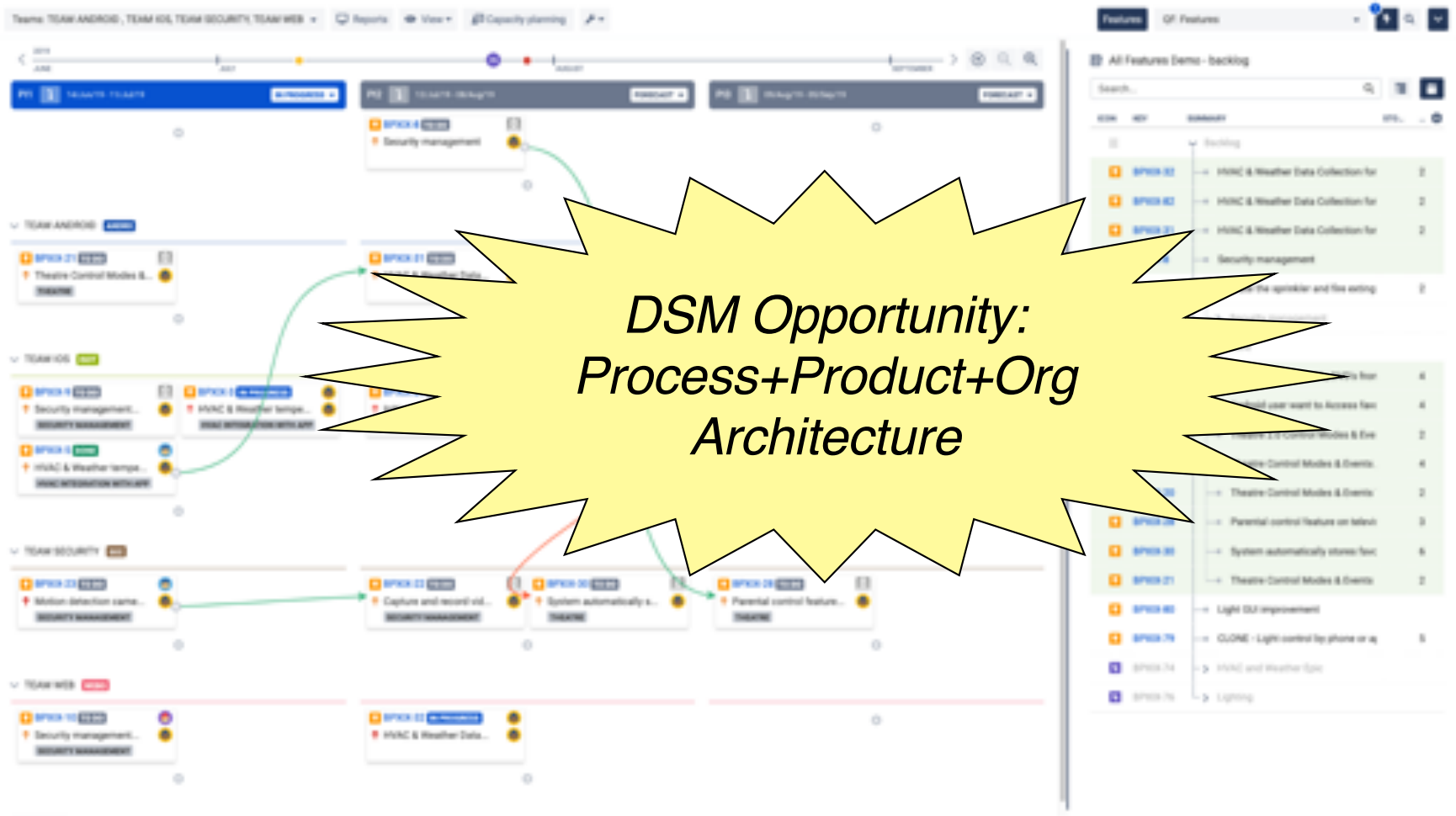
Program Increment



Program Increment (PI) Planning Board: 4x 2-week sprints, 14 teams


Team
Team sprint 1
Team sprint 2
Team sprint 3
Team sprint 4
Team sprint 5
Milestones/events
External dependencies

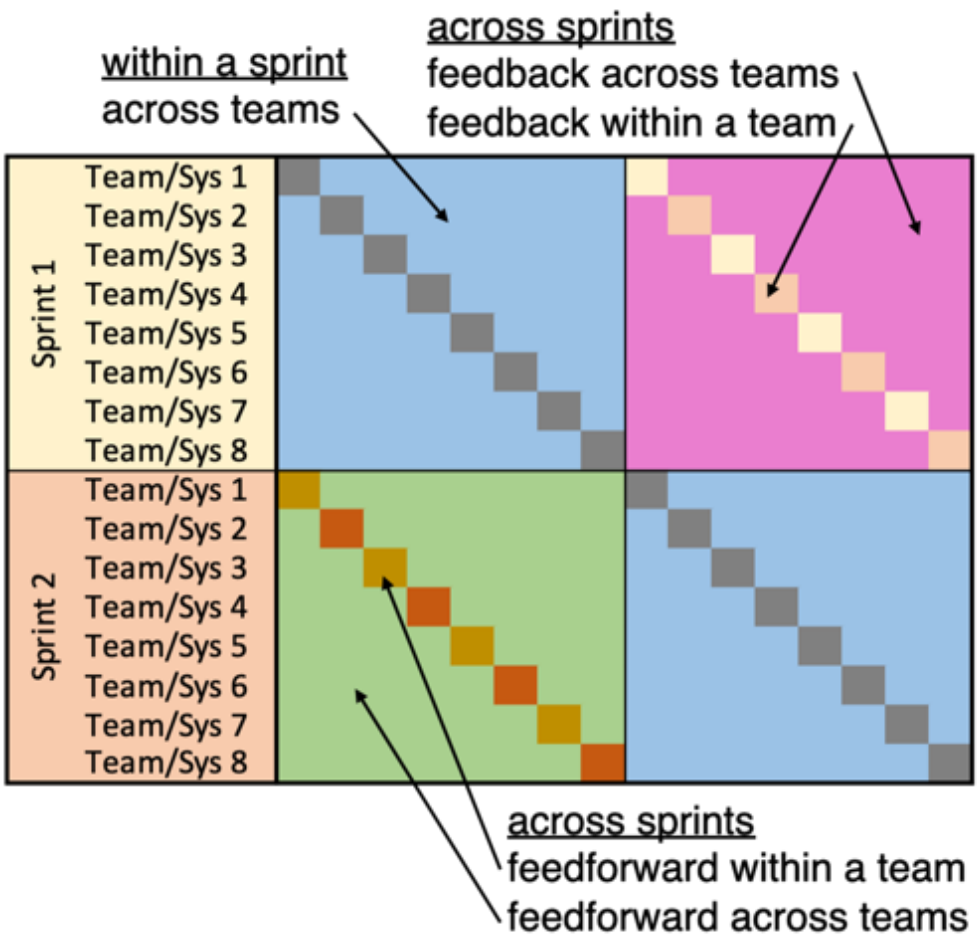
PI Planning Board (Jira plug-in)



https://softwareplant.com/scaled-agile-plugin-jira/

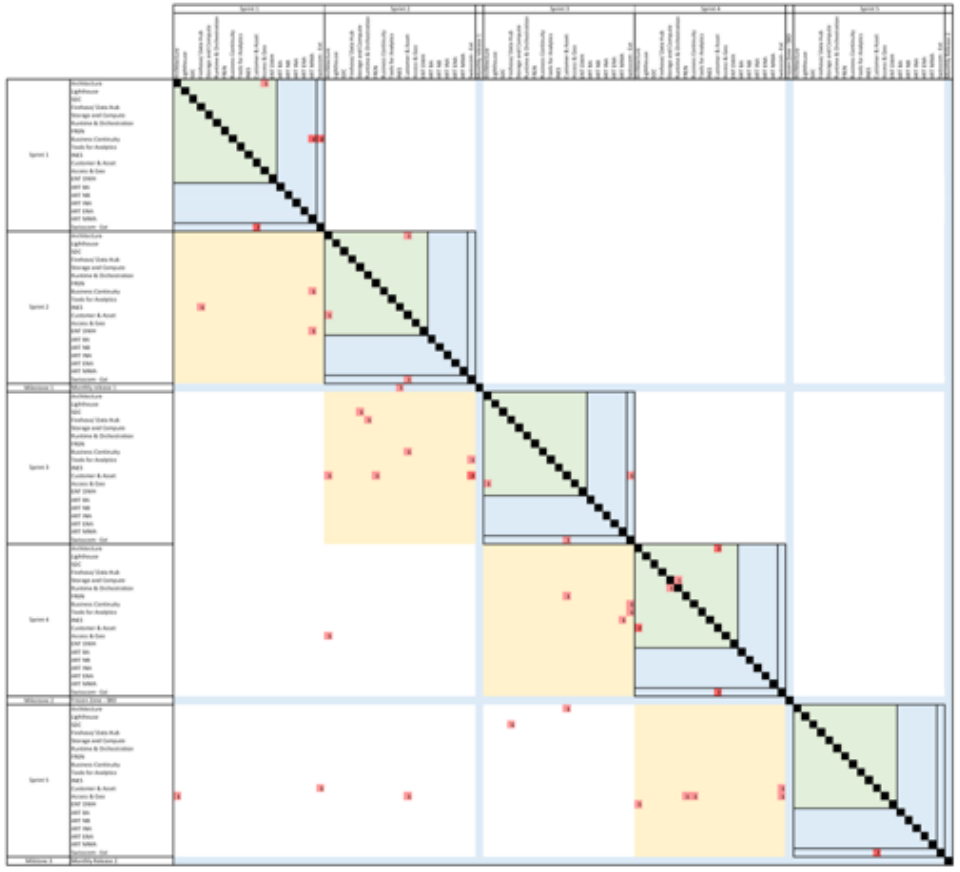
DSM view of PI Planning Board

- First, to give you an idea how this type of DSM might look, here is a little sketch.
- I'm showing you just two sprints in a process DSM. Each sprint includes the work of 8 teams.
- In the blue sections would be the interactions across teams within a sprint.
- The feedforwards (from one sprint to the next) are in the green sections. The pink area is where we'd find any feedbacks.



Feature-Level PI DSM

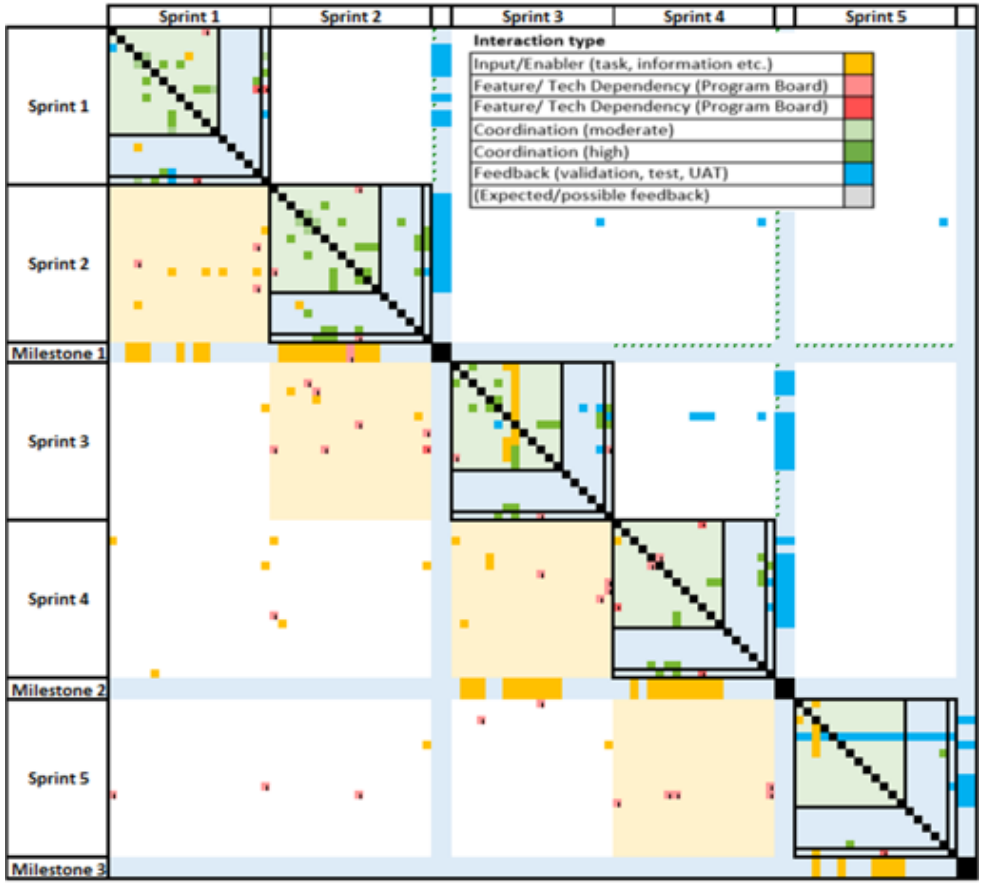
(from program board - during PI planning)



32 interfaces identified on program board

Story-Level PI DSM

(from JIRA data - during PI sprints)



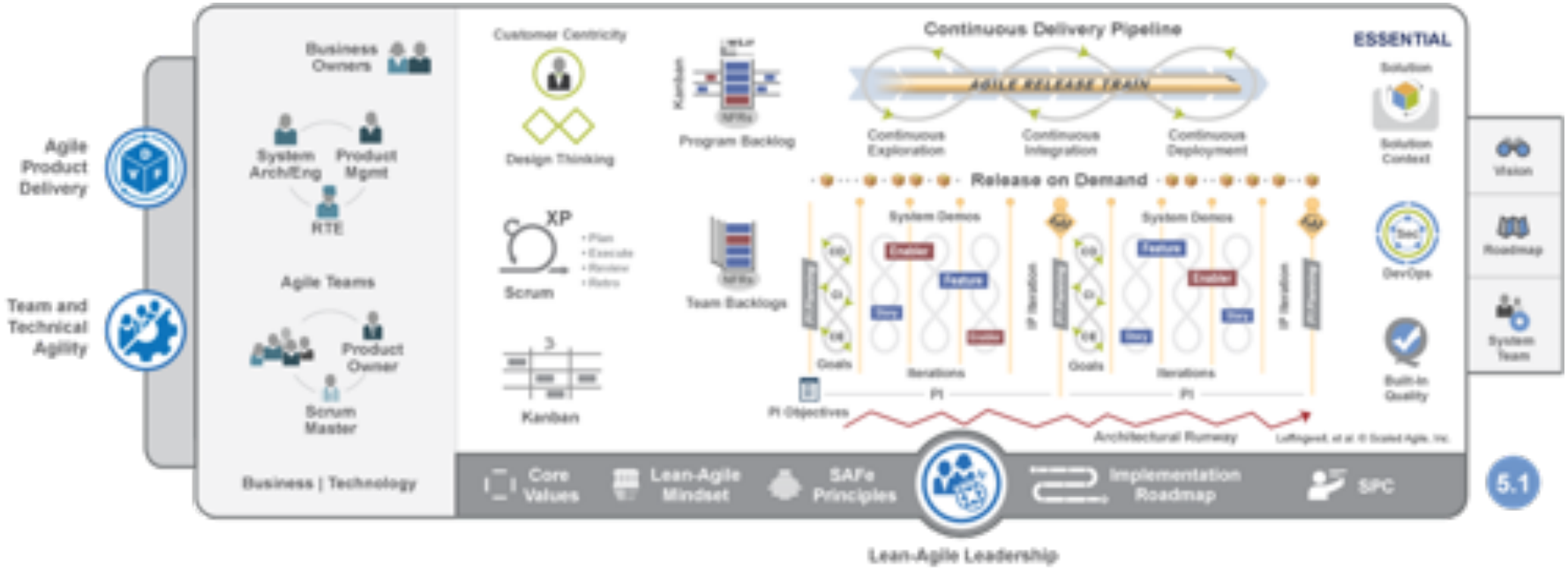
304 interfaces identified in JIRA database

Ref: Enhancing Visibility in Agile Program Increment DSMs, *International DSM Conference, Oct 2020*

swisscom

Essential SAFe includes multiple agile teams working together at the program level.

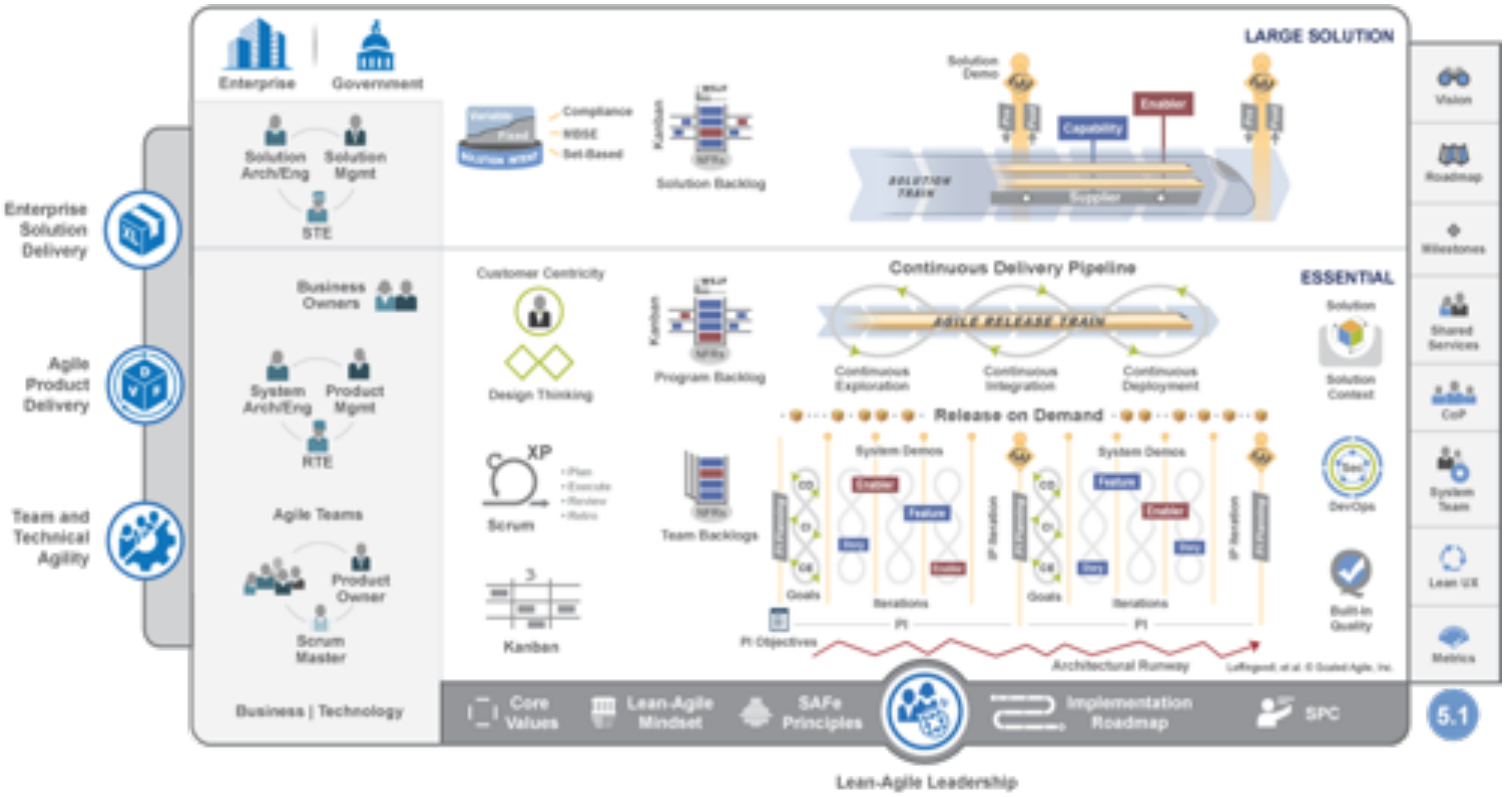
- Several scrum teams form the agile release train (ART).
- Program increments include five sprints (iterations). The 5th sprint is the I&P iteration.



https://www.scaledagileframework.com

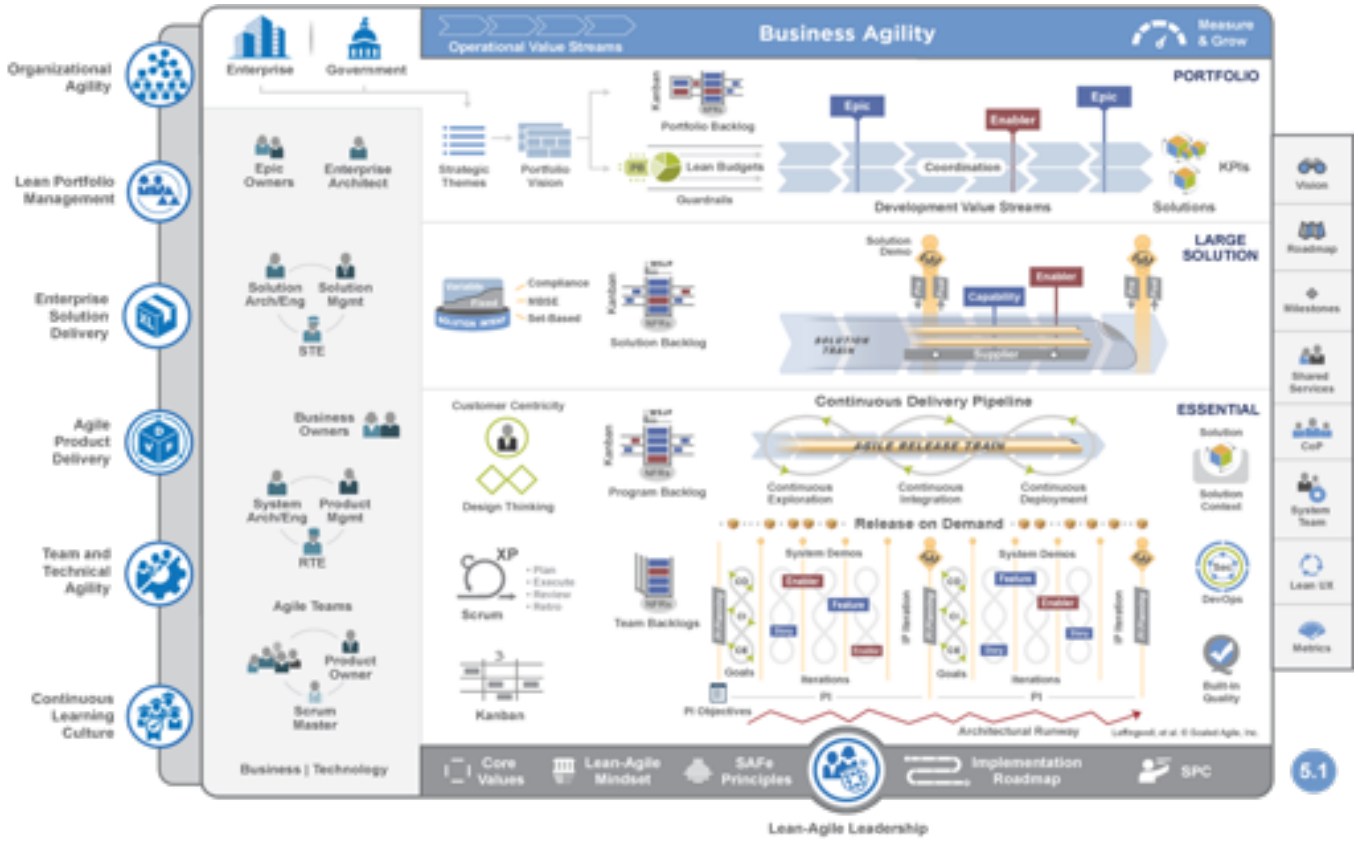
Large Solution SAFe includes multiple program teams working to create a more complex system.

- Large solution is the layer above the program level.
- Includes solution train and roles (solution architect and solution train engineer).

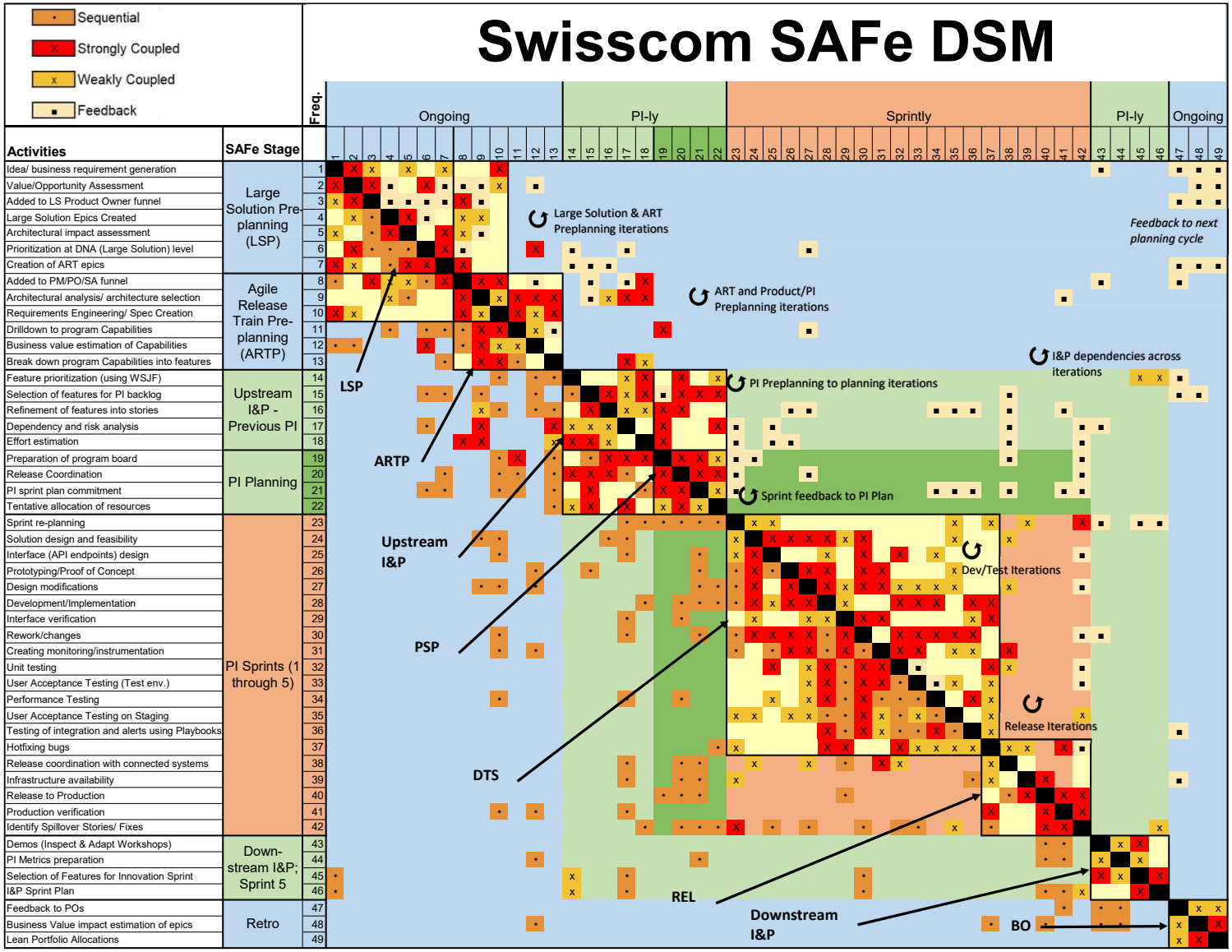


Full SAFe includes governance at the portfolio-level to manage multiple projects and programs.

- Portfolio management layer provides multi-project/program planning.
- The portfolio backlog is prioritized based on overall business strategy.



Swisscom SAFe DSM



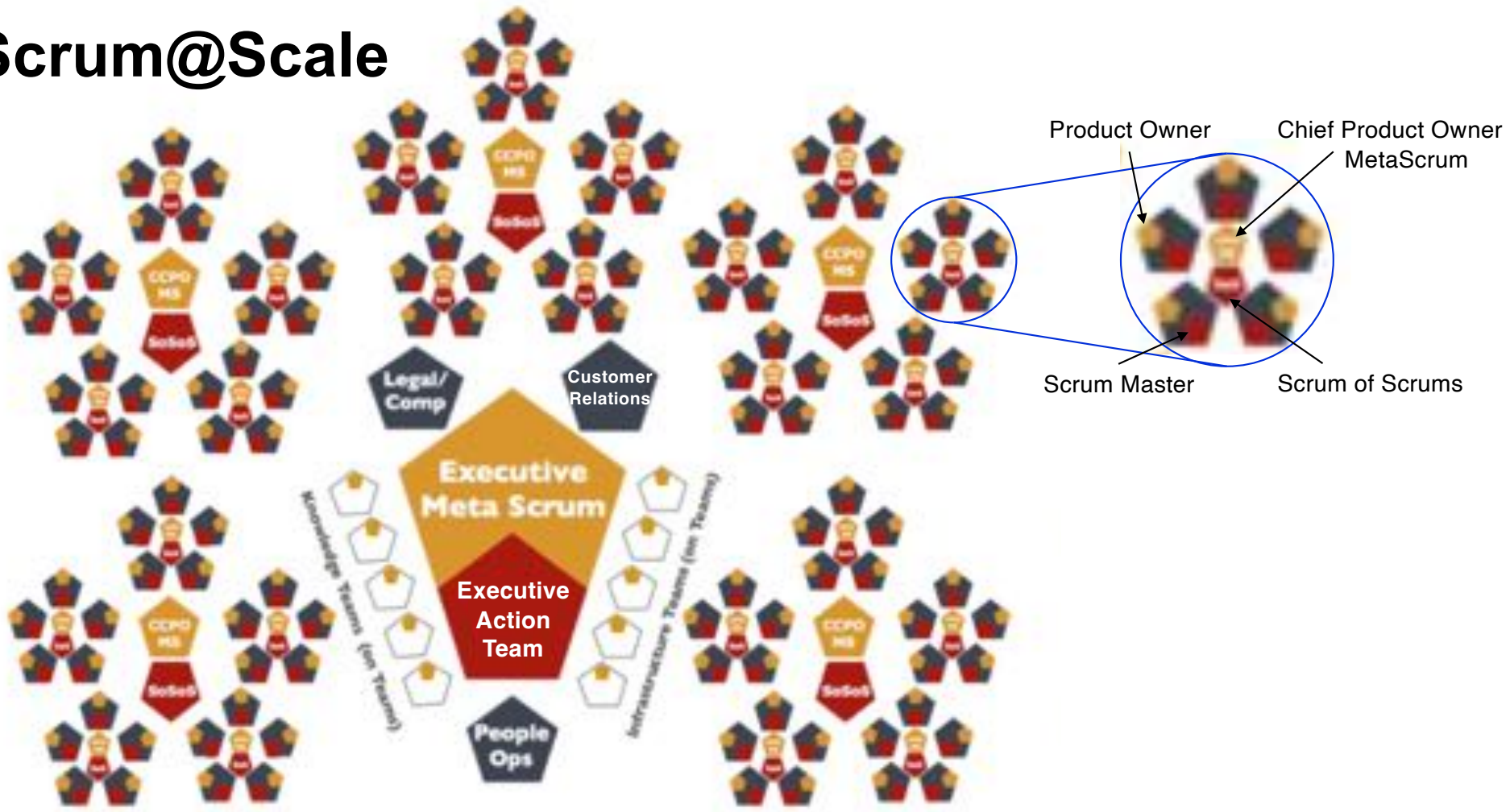
Mature implementation of scaled agile

Layered planning

Nested iterations at various periods

Ref: "The Structure of Agile Development Under Scaled Planning and Coordination", International DSM Conference, Sept 2019.

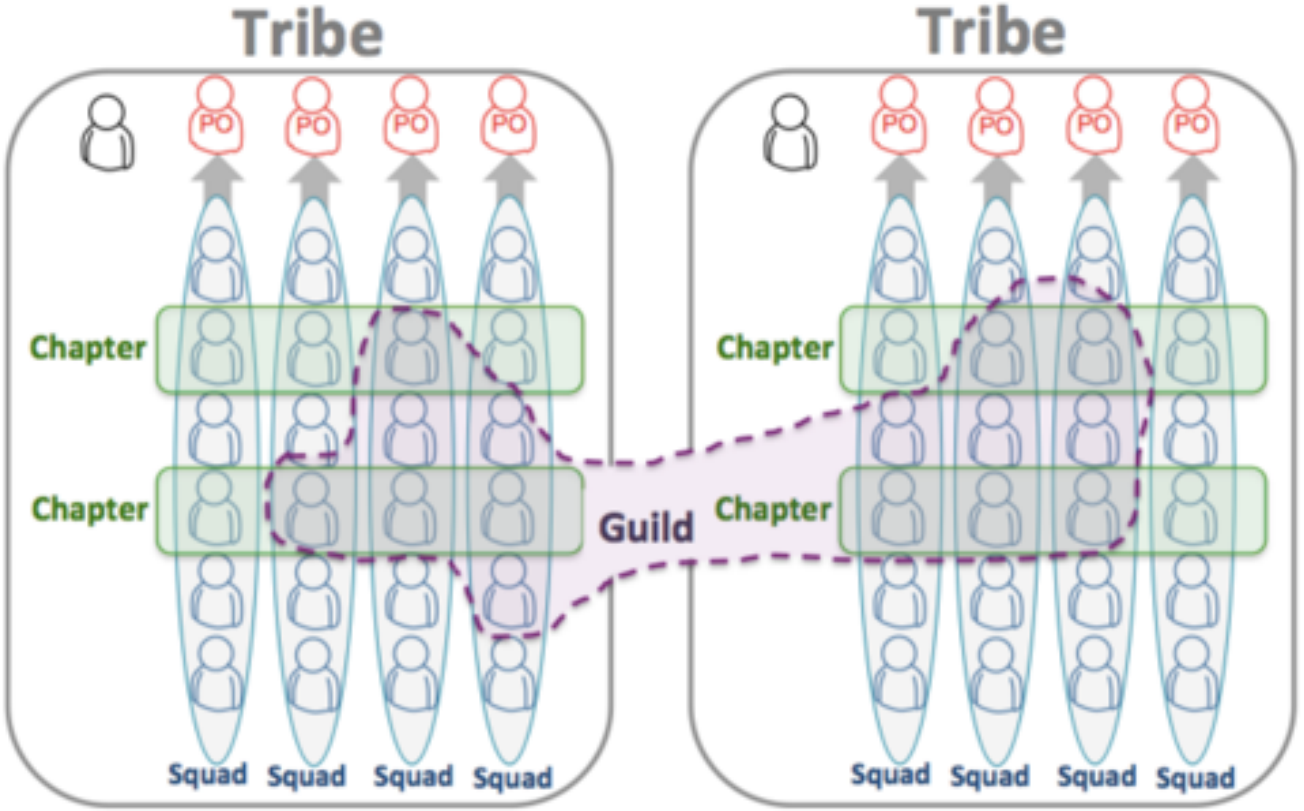
Scrum@Scale



Refs: Jeff Sutherland, *The Scrum@Scale Guide*, 2019
https://www.scrumatscale.com

Spotify uses several layers of cross-team interfaces.

Squad = small agile team
Chapter = similar functions
Tribe = group of squads
Guild = community of practice



Scaling Agile @ Spotify with Tribes, Squads, Chapters & Guilds
Henrik Kniberg & Anders Ivarsson Oct 2012

Agile Transformation

graphic: lithespeed.com

Agile Transformation: My advice in 7 steps

1. ## Start small, then scale up.

Choose a small project to begin with. Use it to help people learn how agile works. The project should be small enough that it doesn't require too many teams or too many external interfaces.

2. ## Choose an easy place to start.

Don't start with a project where agile is hard to do, or one that needs too many adjustments to the standard agile project management approach.

3. ## Try to apply agile right.

While adjustments are often necessary, there are some elements that are critical and they work together – time-boxed sprints, SM and PO roles, backlogs, visual management, and all the sprint events. It will get better over a few sprints.

4. ## Make the effort visible.

By showing off the process (and the results), others will want to learn about agile, and may want to try to work this way. Agile knowledge spreads through other parts of the organization.

Agile Transformation: My advice in 7 steps

5. **You need senior management support.**

Most managers have heard about agile. Explain that they have a role to play too. This includes motivation, goal setting, and shielding agile teams from interruptions.

6. **Get some training.**

Hire an agile coach or an experienced consultant who can do this. You might even have someone in your own software or IT organization who has the right experience.

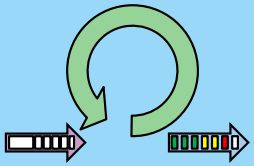
7. **Stay the course.**

This is a long haul. It will take several sprints for each team to get good at this. And it will take several projects (and many teams) to adopt agile before you have a critical mass of experience. Then you're well along your agile transformation journey.

Most Common Challenges

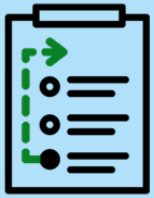
The top three barriers to adopting agile methods (and ways to help)

Short Sprints



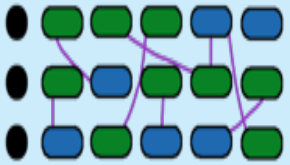
- Embrace the short term
- Break down milestones
- Set a goal for every sprint
- Releases are optional

Backlog Planning



- Develop product owners
- Re-prioritize every sprint
- Engage with stakeholders
- Team refines their PBIs

Managing Interfaces

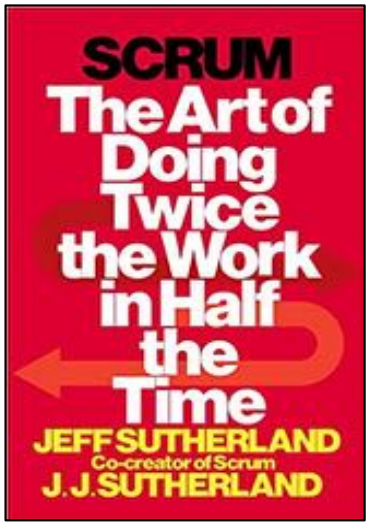


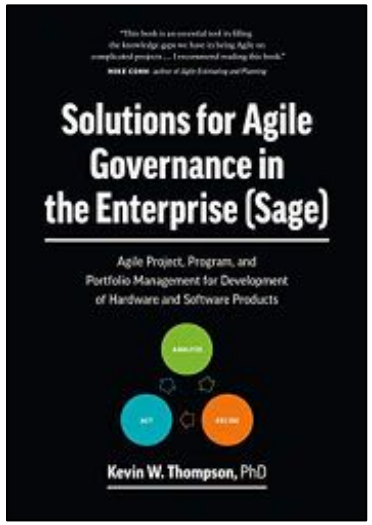
- Plan multiple sprints
- Plan across teams
- Build in modularity
- Interactions can emerge

Agile Transition Coaching:
How to get started with agile

1. Groups include those with and without agile experience.
2. Describe the business process to be considered.
3. How can agile be applied?
   - At the level of teams
   - At the level of product, system, or portfolio
4. Which elements of agile would you apply?
5. What impediments do you anticipate?

Some Agile Reading





- *The Power of Scrum*, Jeff Sutherland, et al., 2011
- *Scrum: The Art of Doing Twice the Work in Half the Time*, Jeff and JJ Sutherland, 2014
- *Solutions for Agile Governance in the Enterprise*, Kevin Thompson, 2019.

10 Agile Ideas You Can Use

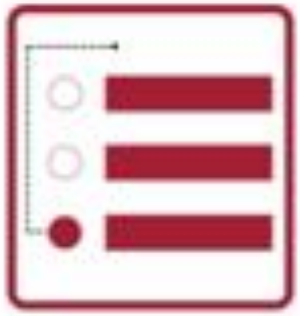
Time-Boxed Sprint

Sprint Planning

Daily Meeting

Sprint Demo/Review

Sprint Retrospective



Product Backlog

Sprint Backlog

Scrum Team

Team Leadership

Agile Values

Please complete your
Program Evaluation
online.

Thank you.

We take your comments very seriously!