MULTI-DOMAIN DSM: SIMULTANEOUS OPTIMIZATION OF PRODUCT, PROCESS & PEOPLE DSMS

Ali A. Yassine
Engineering Management Program, American University of Beirut, Lebanon

Keywords: multi-domain DSMs, partitioning, clustering

1 INTRODUCTION

Product development (PD) is a crucial activity in modern organizations as it fuels their engine of renewal and shapes long-term organizational health and profitability. Engineering organizations continuously introduce new products by executing PD projects using established procedures (i.e., processes), while leveraging a large pool of talented professionals within and across organizational boundaries. As such, an engineering organization can be framed along three orthogonal dimensions (which we refer to as domains): the product to be developed, the activities involved in this development project, and the people entrusted with executing these activities.

As competition stiffens, these organizations are now not only forced to continuously introduce new products, but also to shorten development lead-times, reduce development costs, and improve product variety and quality. Organizations have responded successfully to this challenge by adopting concurrent engineering and lean practices to reduce time and cost, and by quality and platform programs to improve quality and create economic variety. In this paper, we investigate one such tool (or family of tools) called the Design Structure Matrix (DSM). The DSM has been successfully implemented in various improvement scenarios as follows and as summarized in Table 1 (Yassine and Braha 2003).

1. Process modeling for process streamlining in order to reduce time and cost,
2. Product modeling to identify a common platform and modules that promote parallel development of modules and facilitate product variety, and
3. People modeling to compose multifunctional and concurrent engineering teams.

Table 1. Various DSM domains

DSM Domain	DSM Model	Operational Improvements	DSM Analyses	References
Product / System	Product / system DSM	Cross-functional teams, co-located teams	Clustering, real options	Yu et al. 2007; Sharman and Yassine 2004
Process / project	Process / task DSM	Iteration, rework, waste	Partitioning, tearing, banding, simulation	Meir et al. 2007; Browning and Eppinger 2002; Grose 1994
People / Teams	Team / organizational DSM	Platform, commonality, modularity, variety	Clustering, social network analysis	Collins et al. 2009; Pimmler and Eppinger 1994

These various practices and programs has been fruitful over the last two decades, but have reached the efficient frontier in terms of operational effectiveness, where it is difficult to achieve further improvements within a single domain. As these local inefficiencies have been dealt with successfully, system inefficiencies (that lie at the intersection of the above three domains) present a viable opportunity for further operational improvements.

In this paper, we introduce domain-spanning rules that can be used in the construction of mathematical objective functions in order to optimize DSMs across multiple domains.

2 DSM PRELIMINARIES: OPTIMIZING SINGLE-DOMAIN DSMS

In this section we present a quick overview of various rules (i.e. objectives functions) utilized in the optimization of single-domain DSMs. Therefore, highlighting the need for cross-domain analyses and the development of domain-spanning rules used in multi-domain DSM optimization.

2.1 DSM partitioning & clustering

A DSM is a binary square matrix that shows process elements (or tasks) along its diagonal, activity outputs in its columns and activity inputs in its rows. Thus, feed-forward information in the process is shown below the diagonal, and non-zero super-diagonal cells indicate the presence of feedback. The design process can be sequenced or partitioned (i.e., the information flow streamlined) by reordering the activities, such that feedback, and thus iteration, is minimized (Steward 1981). A simple metric or objective function is to choose the DSM arrangement that minimizes the total count of feedback marks, as shown in the first row of Table 2. A more sophisticated objective function would take into consideration the lengthy of the feedback loop as shown in the second row of Table 2. More involved objectives for DSM partitioning could also be devised (Meir et al. 2007).

Table 2. DSM partitioning and clustering objective functions

Objective	Rationale	Objective Function Equation
Minimize number of feedback marks (Steward 1981)	Reduce iteration	$f = \sum_{i=1}^{n} \sum_{j=i+1}^{n} w(i,j)$ where $w(i,j)$ is a binary value in cell (i,j)
Minimize total feedback length; i.e. Minimize distance from diagonal (Gebala and Eppinger 1991)	Reduce iteration length	$f = \sum_{i=1}^{n} \sum_{j=i+1}^{n} (j-i) \cdot w(i,j)$ where $w(i,j)$ is a weighted value bet. 0 &1 in cell (i,j)
Minimize the sum of internal (i.e. within) and external (i.e. between) coordination cost between various clusters (Thebeau 2001)	Reduce coordination cost	$f = \sum_{i=1}^{c} (I_i)(n_i)^{\alpha} + \sum_{i=1}^{c} (I) N^{\beta}$ where I_i is the sum of marks within cluster i, I is the sum of marks outside all clusters, c is the number of clusters in the DSM, N is the total number of nodes in the DSM, n_i is the number of nodes in the i^{th} cluster, and α and β are weights ≥ 1.
Clustering – Improve modularity (Yu et al. 2007)	Information theoretic measure of fitness	$f = (1-\alpha-\beta) \cdot \left(c \log N + \log N \sum_{i=1}^{n_c} n_i \right) + \alpha \cdot [\lvert S_1 \rvert (2 \log N + 1)] + \beta \cdot [\lvert S_2 \rvert (2 \log N + 1)]$ where c, N and n_i are same as above, and α and β are weights between 0 and 1.

When the DSM elements are people in charge of tasks or are sub-systems and components of a larger system, then the overarching objective for arranging the DSM is to cluster the DSM. Clustering is the problem of finding subsets of DSM elements (i.e., clusters or modules) that are mutually exclusive or minimally interacting (Sharman and Yassine 2004). Furthermore, in this setting, marks below the diagonal are synonymous to marks above the diagonal and they represent interactions between the teams or interfaces between the modules. Mathematically, clustering can also be achieved using several objective functions. The simplest is when we consider the differential count between the number of marks within and outside any clustering arrangement as shown in the third row of Table 2. More sophisticated objective functions were devised based on the minimum description length as shown in the last row of Table 2 (Yu et al. 2007).

2.2 Relevant multi-domain DSM literature

Matrix mapping approach, in general, between various interrelated domains is discussed thoroughly by Yassine et al. (2003). Eppinger and Salminen (2001) discuss the possible relationship between the product, tasks, and organizational domains in product development. Sharman et al. (2002) suggest that elements in one domain need to map to the same element in another domain in a one-to-one manner. They propose a hypothetical optimization of a multiple-domain PD project resulting in an optimal Design Structure Matrix (DSM) showing the relational arrangement of elements in the various domains. Danilovic and Browning (2007) propose a rectangular DSM construction relating DSM's representing different domains of the product development process. This new domain mapping matrix (DMM) provides insights into the various characteristics of the product development process. Finally, Maurer et al. (2007) considers a multi-domain approach that considers the complexity cycle for multiple factors including market complexity, product complexity, process complexity, and organizational complexity. They propose a scheme that relates these domains by elements of information sharing activity taking place within the organization. These multiple domain elements map to a new multi-domain network.

However, what is missing from all the above research, and what this paper offers, is the lack of mathematical constructs, similar to those presented in Table 2 for single-domain DSMs, which can be used in optimizing multi-domain DSMs. Thus, in this paper we propose a set of domain-spanning rules that lay the ground work for mathematical optimization of multi-domain DSMs.

3 OPTIMIZING MULTI-DOMAIN DSMS

Local or individual domain optimization requires mapping the relationships in any one of the three domains and running the proper analysis procedure (using the proper objective function) to select an optimal arrangement for that specific domain. This could be repeated for all three domains yielding three individual local optimal solutions. However, for a global solution, this requires the optimization of all three domains simultaneously; perhaps using a single objective function that cuts across all three domains. This is best illustrated with a simple example showing how two domains can possibly interact.

The domains chosen are the process/task and product/physical. Assume one-to-one mapping between elements in process domain and elements in the product domain. Also, assume that sequence of elements in one domain forces the same sequence in the other domain. As such, when the information in these two domains is combined, then the sequence and module boundary shown in Figure 1 represent a good compromise. This is relatively harmonious as the B:C module or subsystem can be developed independent of the D:G module provided the B→G relationship is respected (the reason the task domain drives forwards from B to G is a result of deliberate selection; there may indeed have been an alternative solution that allowed G→B to drive the design). However, if the information contained in the physical domain had not been known, then the information flows in the process domain could easily have yielded Figure 2. If this had happened it would be impossible to separate out two modules in the physical domain as the sequence of choosing inter-element interfaces makes it unlikely that they would decompose cleanly. This is akin to the situation faced by poorly understood products as the physical manifestation is forced to be more integrated than it need be.

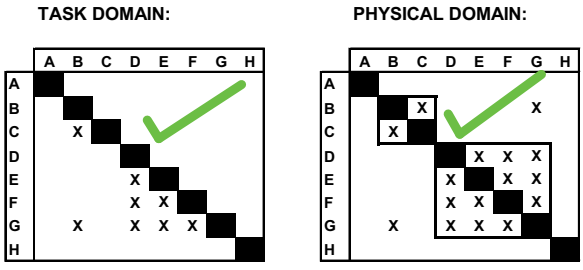


Figure 1. Good optimization of two domains

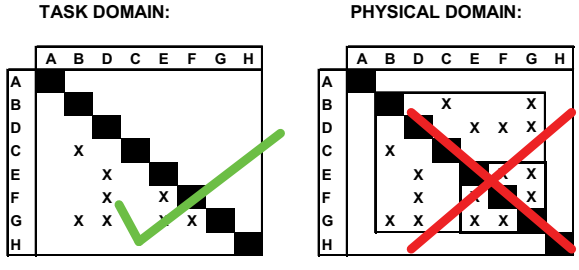


Figure 2. Poor optimization from process domain focus

One major assumption in the above argument is that the arrangement of one DSM domain must mirror the other 2 domains, which is not necessarily true. If we allow elements is different domains to vary but keeping a common thread of relationship between them, we may be able to discover more efficient multi-domain DSM arrangements that were not originally apparent. For example, if two elements in the product domain belong to the same cluster, then it may be necessary (or beneficial) that their counterpart in the people domain belong to the same team. Similar rules can be devised based on our understanding of the impact of one domain (e.g., its arrangement) on the optimal arrangement of other domains. A set of such rules is proposed in Table 3.

Table 3. Proposed multi-domain rules

Rule	Description
Team →Process	If 2 (or more) tasks are in feedback and they are performed by the same resource or by resources that belong to the same team, then the feedback penalty is removed or at least reduced.
Process →Product	If (2 or more) product modules are related through various interfaces and the tasks corresponding to these interfaces are sequential (i.e., not involved in feedback), then the module interface penalty is removed or at least reduced.
Product → Team	If 2 (or more) teams work on disjoint product modules, then the interface penalty between these teams is removed or at least reduced.

The proposed multi-domain optimization may proceed as follows. First, we identify individual domain optimizations, and then a global optimum is constructed by qualitatively identifying areas of tension between domains, mostly captured based on experience (Parashar & Bloebaum 2005). The generic multi-domain optimization objective function may be of the following form, where w_j represent the coefficients of the j'th domain's contribution to value:

$$\textbf{\textit{Objective Function: Max.}}\ \sum_i \{w_i \,.\, (\textit{Value of domain } i \pm \textit{Influence of } j^{th} \textit{ domain on domain } i)\} \quad (1)$$

4 Example

Consider the simple 2-domain example shown in Figure 3. Again, assume that the two domains are the team DSM and the process DSM. We assume that team A is related to process 1, team B is related to process 2, and Team C & D is related to process 3. The calculations involved in solving this two-domain problem utilize Equation (1). The domain value portion of Equation (1) is based on the first and third rows of Table 2. The "influence" portion of Equation (1) is based on the first rule in Table 3. Finally, the domain contribution to value (i.e. weights w_1 and w_2) are set to 1. A simple Java code was written to automatically generate all 54 two-domain problems and evaluate the corresponding objective function. The arrangement with the minimum objective function value is selected. Table 4 presents sample calculations for 3 of these cases.

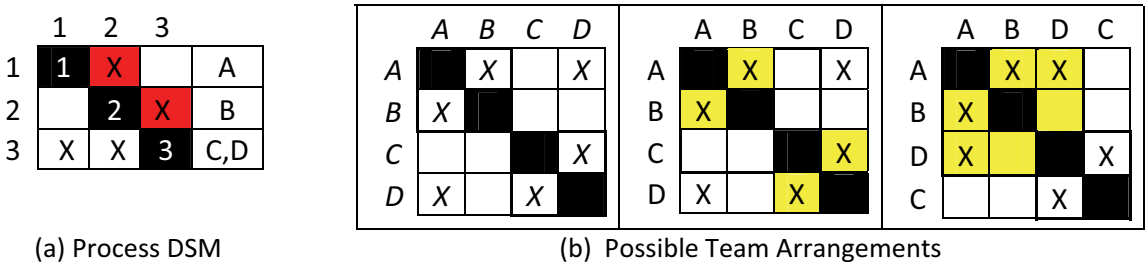


Figure 3. Two-domain example

Table 4. Example calculations for 3 cases

Process	Team	OF(Process)	OF(Team)	Rule 1	Total OF
123	A-B-C-D	2/4 = 0.5	1	No influence	1.5
123	AB-CD	0.5	{1(2)+1(2)+1(4)}/3(4) = 0.67	2→1 feedback: -1/4 = -0.25	0.92
123	ABD-C	0.5	{2(3)+(0(1)+1(4)}/3(4) = 0.83	Both feedbacks: -2/4 = -0.5	0.83

5 SUMMARY AND CONCLUSION

The main point of this paper is to demonstrate that multi-domain DSM optimizing can provide different results from individual domain optimizations. However, it is not a simple matter to develop global objective functions (O.F.) to do so. In general, the best way to develop such global O.F.s is to start by understanding cross domain relationships and devising rules for such relationships. One such relationship deployed in this paper is between the process and team DSMs, which states that whenever there is a feedback between process DSM elements, it is beneficial to have the these feedback elements represented in the same team formations in the corresponding team DSM.

REFERENCES

Browning, T. & Eppinger, S. (2002). Modeling impacts of process architecture on cost and schedule risk in product development. *IEEE Trans. on Eng. Management*, 49(4), 428-442.

Danilovic, M. & Browning, T. (2007). Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management*, 25, 300-314.

Eppinger, S. & Salminen, V. (2001). Patterns of product development interactions. *Proceedings of International Conference on Engineering Design*, ICED'01, August 2001.

Gebala, D.A. & Eppinger, S.D. (1991). Methods for analyzing design procedures," Proceedings of the ASME Third International. Conference On Design Theory and Methodology, September, Miami, FL.

Grose, D.L. (1994). Reengineering the aircraft design process. *Proceedings of the Fifth AIAA/ USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Panama City Beach, FL, Sept. 7-9.

Maurer, M. & Lindemann, U. (2007). Facing multi-domain complexity in product development. Competence in design and development. January 2007.

Meier, C., Yassine, A., & Browning, T. (2007). Design process sequencing with competent genetic algorithms. *Journal of Mechanical Design*, 129(6), 566-585.

Parashar, S. & Bloebaum, C. (2005). Decision support tool for multidisciplinary design optimization (MDO) using multi-domain decomposition. *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, Austin, Texas, 18-21 April 2005.

Pimmler, T. & Eppinger, S.D. (1994). Integration analysis of product decompositions. *Proceedings of the ASME International Conference on Design Theory and Methodology*, DE-68, 343-351.

Sharman, D. & Yassine, A. (2004). Characterizing complex product architectures. *Systems Engineering Journal*, 7(1), 35-60.

Sharman, D., Yassine, A., & Carlile, P. (2002). Architectural optimisation using real options theory and dependency structure matrix. *ASME 2002 International Design Engineering Technical Conferences*. 28th Design Automation Conference, Montreal, Canada, September 29-October 2, 2002.

Steward, D.V. (1981). *Systems Analysis and Management: Structure, Strategy and Design*. New York, NY: Petrocelli Books.

Thebeau, R. (2001). *Knowledge Management of System Interfaces and Interactions for Product Development Processes*. Master Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.

Yassine, A. & Braha, D. (2003). Four complex problems in concurrent engineering and the design structure matrix method. *Concurrent Engineering Research & Applications*, 11(3), 165-176.

Yassine, A., Whitney, D., Daleiden, S., & Lavine, J. (2003). Connectivity maps: Modeling and analyzing relationships in product development processes. *Journal of Engineering Design*, 14(3).

Yu, T., Yassine, A., & Goldberg, A. (2007). Developing modular product architectures using genetic algorithms. *Research in Engineering Design*, 18(2), 91-109.

Contact: Ali A. Yassine
American University of Beirut
Engineering Management Program
Bliss Street
PO Box 11-0236
Beirut
Lebanon
00-961-1-350-000 Extn. 3439
ali.yassine@aub.edu.lb

Multi-Domain DSM: Simultaneous Optimization of Product, Process & People DSMS

Ali Yassine
Engineering Management Program
American University of Beirut
Beirut, Lebanon

Technische Universität München

UNIVERSITY OF CAMBRIDGE

MANAGING COMPLEXITY
BY MODELLING DEPENDENCIES

Outline

- Single-Domain DSM Optimization: Partitioning & Clustering
- Need for Multi-Domain Optimization
- Multi-Domain Rules: Domain Spanning Rules
- Example
- Summary & Conclusion

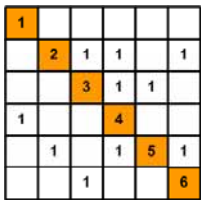
Single Domain Optimization

DSM Domain	DSM Model	Operational Improvements	DSM Analyses	References
Product / System	Product / system DSM	Cross-functional teams, co-located teams	Clustering, real options	Yu et al. 2007; Sharman and Yassine 2004
Process / project	Process / task DSM	Iteration, rework, waste	Partitioning, tearing, banding, simulation	Meir et al. 2007; Browning and Eppinger 2002; Grose, 1994
People / Teams	Team / organizational DSM	Platform, commonality, modularity, variety	Clustering, social network analysis	Collins et al. 2009; Pimmler and Eppinger 1994

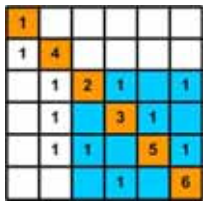
DSM Partitioning and Clustering Objective Functions

Objective	Rationale	Objective Function Equation
Minimize number of feedback marks (Steward 1981)	Reduce iteration	$f = \sum_{i=1}^{n} \sum_{j=i+1}^{n} w(i,j)$ where $w(i,j)$ is a binary value in cell (i,j)
Minimize total feedback length; i.e. Minimize distance from diagonal (Gebala and Eppinger 1991)	Reduce iteration length	$f = \sum_{i=1}^{n} \sum_{j=i+1}^{n} (j-i) \cdot w(i,j)$ where $w(i,j)$ is a weighted value bet. 0 &1 in cell (i,j)
Minimize the sum of internal (i.e. within) and external (i.e. between) coordination cost between various clusters (Thebeau 2001)	Reduce coordination cost	$f = \sum_{i=1}^{c} (I_i)(n_i)^{\alpha} + \sum_{i=1}^{c} (I) N^{\beta}$ Where I_i is the sum of marks within cluster i, I is the sum of marks outside all clusters, c is the number of clusters in the DSM, N is the total number of nodes in the DSM, n_i is the number of nodes in the i^{th} cluster, and α and β are weights ≥ 1.
Clustering – Improve modularity (Yu et al. 2007)	Information theoretic measure of fitness	$f = (1-\alpha-\beta) \cdot \left(c \log N + \log N \sum_{i=1}^{n_c} n_i \right) + \alpha \cdot [\lvert S_1 \rvert (2 \log N + 1)] + \beta \cdot [\lvert S_2 \rvert (2 \log N + 1)]$ where c, N and n_i are same as above, and α and β are weights between 0 and 1.

Simple Partitioning Objective Function (OF)

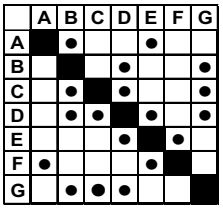


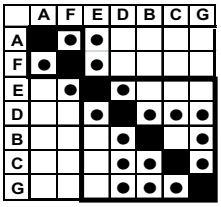
OF1 = 6/10=0.6



OF1 = 4/10=0.4

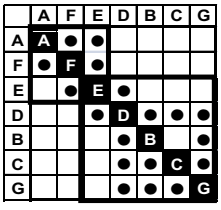
Simple Clustering Objective Function (OF)





OF3 = 1(2)+7(5)
+1(7) = 44

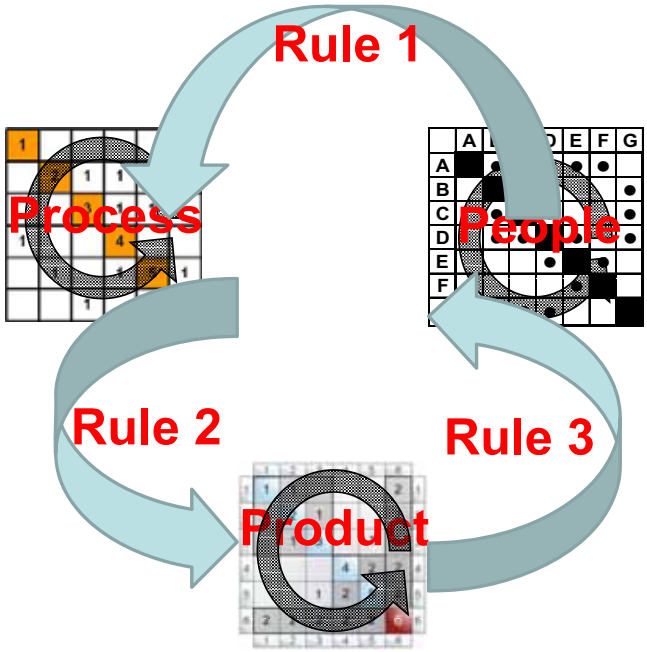
= 44/(9*7)
= 0.7



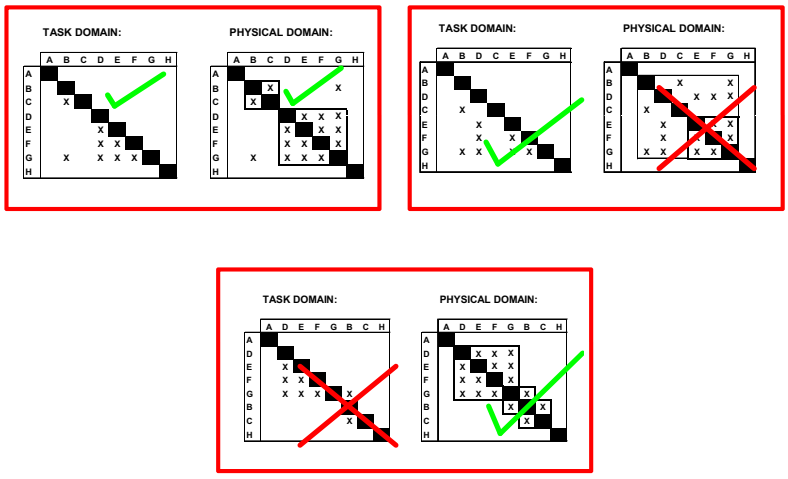
OF3 = 2(3)+7(5)
+0(7) = 41

= 41/(9*7)
= 0.65

Multi-Domain DSM Optimization



Simple Examples Showing Multi-DSM Influences



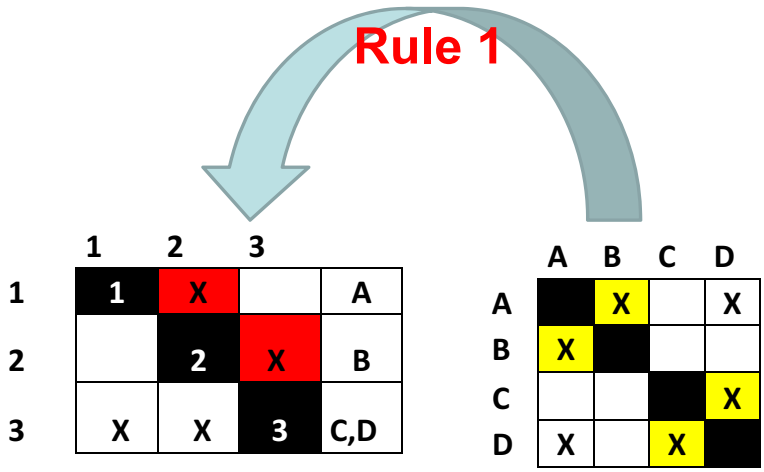
Multi-Domain Rules

Rule	Description
Team →Process	If 2 (or more) tasks are in feedback and they are performed by the same resource or by resources that belong to the same team, then the feedback penalty is removed or at least reduced.
Process →Product	If 2 (or more) product modules are related through various interfaces and the tasks corresponding to these interfaces are sequential (i.e., not involved in feedback), then the module interface penalty is removed or at least reduced.
Product → Team	If 2 (or more) teams work on disjoint product modules, then the interface penalty between these teams is removed or at least reduced.

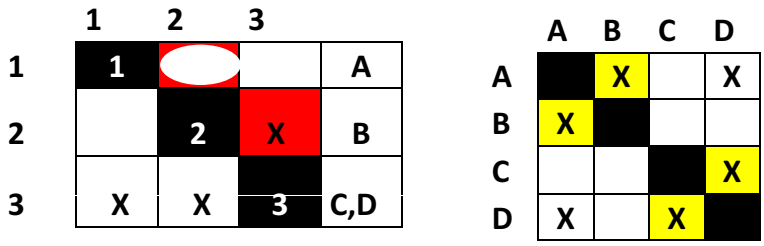
Multi-Domain Objective Fiunction (OF)

Objective Function:
Max $\sum_i \{w_i$ ***. (Value of domain i ± Influence of*** j^{th} ***domain on domain i)\}$

Two-Domain Optimization



Two-Domain Optimization



Two-Domain Optimization

	1	2	3	
1	1			A
2		2		B
3	X	X	3	C,D

	A	B	D	C
A		X	X	
B	X			
D	X			X
C			X	

Two-Domain Optimization

Rule 3

	A	B	C	D	Product Clusters
A		X		X	M1
B	X				M1,M2
C				X	M2
D	X		X		M2

	a	b	c	d	Tasks	Resources
a				X	1,2	A,B
b					1	A
c					3	C,D
d	X				2	B

Summary & Conclusion

- Multi-domain DSM optimizing can provide different results from the linear sum of individual domain optimizations.
- It is not a simple matter to develop global objective functions (O.F.). In general, the best way to develop such global O.F.s is to start by understanding cross domain relationships and devise domain-spanning rules accordingly.
- Three domain-spanning rules were suggested; however, these rules can be improved.
- Exhaustive enumeration and O.F. evaluation for Large multi-domain problems is prohibitive and requires some sort of meta-heuristic. Next steps.