

# Evaluating Product Development Task Interactions Using Network Analysis

Shawn Collins  
Department of Anthropology  
University of Connecticut  
Storrs, CT. 060XX  
[scollins@alumni.purdue.edu](mailto:scollins@alumni.purdue.edu)

Ali A. Yassine  
Dept. of Industrial & Enterprise Systems Eng.  
University of Illinois at Urbana-Champaign  
Urbana, IL. 61801  
[yassine@uiuc.edu](mailto:yassine@uiuc.edu)

## Abstract

This paper proposes the integration of two useful systems engineering analysis tools, the Design Structure Matrix (DSM) and Network Analysis (NA) to study task interactions in a Product Development Process (PDP) using a small engineering company as a case study. The DSM is a matrix-based systems engineering tool that enables streamlining of the PDP through task sequencing and simulation of project execution. NA techniques such as key players, centrality, influence, and brokerage provide methods to identify critical product development tasks and interactions to focus PDP improvement. Collecting interactions between product development tasks in matrix form provides opportunities to use DSM and NA techniques to identify highly central tasks and identify patterns of cross-coupling between tasks and the groups performing those tasks. One benefit is the ability to describe feedback characteristics of critical PDP tasks. This is valuable when evaluating the impact of omitting, combining, or re-sequencing task execution based on program-specific constraints.

## Overview

The product development process (PDP) is a complex system composed of an integrated set of tasks that collectively accomplish a defined objective – e.g. developing a new product (INCOSE 2004, Browning et al. 2006). As such, many existing systems engineering tools can be used to improve our understanding and analysis of these kinds of systems (i.e. product development processes). Graph-based techniques (such as CPM/PERT, IDEF and flowcharts) and matrix-based techniques (such as DSM and  $N^2$  method) were successful in capturing the relationships between the tasks in a PDP and in scheduling these tasks accordingly (Browning 2001, Bustnay and Ben-Asher 2005). However, almost all these approaches provide a partial view of the development process because they lack the sophistication for uncovering the underlying statistical properties of the development process and its constituent tasks.

We can enhance our understanding of these systems by using established network analysis (NA) techniques, which help uncover previously unnoticed trends and properties in the product development process (PDP). For example, it is possible to look at patterns of the overall PDP structure and the location of individuals within this structure. This is an important analytical shift from viewing a task as having individually unique characteristics (as is the case in the above mentioned SE tools) to viewing it as representing emergent properties - emerging from patterns of relationships between tasks. In particular, using NA techniques to measure

various task properties (such as key players, centrality, influence, and brokerage) provide a methodology to identify critical product development tasks and interactions to focus PDP improvements.

This paper uses the Design Structure Matrix (DSM) and Network Analysis (NA) techniques to analyze task interactions in the Product Development Process (PDP) at a small engineering company (Smallcomp). Collecting interactions between product development tasks in matrix form provides opportunities to use DSM and NA techniques to identify highly central tasks and identify patterns of cross-coupling so that impacts between tasks (and the groups performing those tasks) are better understood. A significant benefit is the ability to describe feedback characteristics of critical product development tasks. This is especially valuable when evaluating the impact of omitting, combining, or re-sequencing task execution based on program-specific constraints.

The rest of the paper proceeds as follows. In the next section we introduce the Design Structure Matrix (DSM). The third section discusses Network Analysis (NA) data format, display, and key concepts that can be used to analyze interactions among Product Development tasks. The fourth section applies several NA concepts to DSM data on Smallcomp's PDP. This analysis provides insight into critical interactions between key PDP tasks at the overall process, individual design phase, and specific task levels. The last section discusses lessons learned and applications.

## The Design Structure Matrix

The Design Structure Matrix (DSM) is a system engineering tool that uses matrices to model and analyze complex projects, processes or systems (Browning 2001, Yassine and Braha 2003). The collected data in a DSM captures the structure of interactions, interdependencies and interfaces between physical system elements like subsystems and modules, product development teams and information flow between project tasks. *Static DSM* analysis uses clustering algorithms to identify groups of close-coupled components to identify sub-systems or frequently interacting individuals to identify project teams (McCord and Eppinger 1993, Sosa, et al. 2003, 2004) or product modules (Yu et al. 2003, Sharman and Yassine 2004). *Time-based DSM* analysis uses partitioning algorithms that optimally sequence project tasks to minimize the impact of rework and unexpected feedback (Yassine et al. 2003). The information about task cross-coupling, re-work impact, and re-work probability identifies a range of possible project durations to assist in planning and resource allocation. This inclusion of interdependencies and iteration is an improvement over the assumption of linearly independent, sequentially ordered tasks in time-based planning tools like MS Project and sequence analysis tools like Value Stream Mapping (Tapping and Shuker 2003).

Figure 1 shows the two different approaches to analyzing DSM data. The strength of interaction is a two-point scale of "Does not interact" (matrix cell is empty) and "Output has some influence" (matrix cell value of 1). The time-based data has potential for re-work because the outputs from tasks G and F provide input to earlier chronological tasks E, C and B. The partitioned DSM has re-ordered the task execution sequence so that the only feedback loop is between tasks G and F.

One criticism of project management using the DSM is that the task interaction matrix assumes that feedback to earlier tasks is undesirable. This is not true if activities are concurrent or intentionally iterative, as is the case in research and development environments. An alternative approach to partitioning is to cluster the matrix into task subsets that are mutually exclusive or minimally interacting. Static DSM analysis uses this approach for data which

typically do not contain time dependent information, such as hardware components or team responsibilities. In principal it can be used to identify groups of R&D tasks that are intentionally iterative or concurrently scheduled. A weakness of this approach for rigorous understanding of R&D projects is that DSM techniques currently do not include methods to evaluate the within and between group tie characteristics once clusters are identified. However, a number of diagnostic metrics are available from Network Analysis (NA).

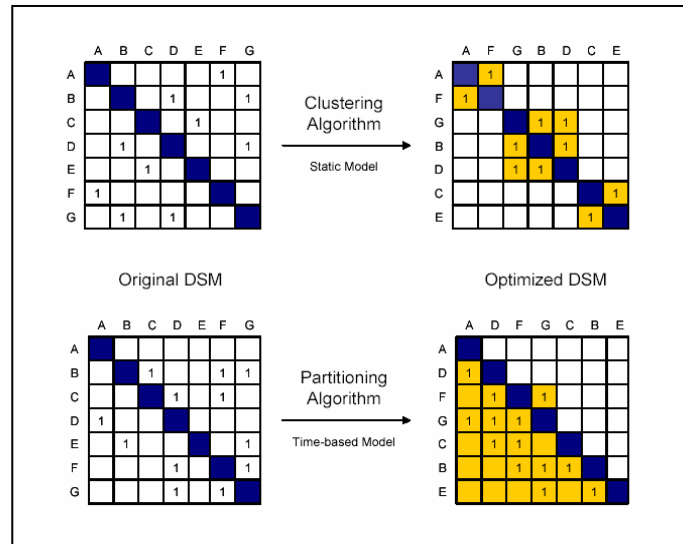


Figure 1: Static and Time-Based DSM Analysis

## Network Analysis Concepts for Process Management

Collecting data in network form and applying social network metrics has a number of conceptual benefits for evaluating complex systems (Scott 2000, Kilduff and Tsai 2003, Batallas and Yassine 2006). First, network analysis focuses on relations and patterns of relations rather than attributes of actors or tasks. Second, it is amenable to multiple levels of analysis, allowing insight into micro-macro linkages. For example, while an overall network influences patterns of individual members, individual actions create sub-groups that exert influence for a particular type of relationship. The network approach thus provides insight into how individuals affect institutions, and how institutions constrain individuals (e.g. CEO's and dominance over distribution of resources).

One major benefit is that the overall structure tends to be emergent, in that no individual in the network understands the entire system. Network analysis uses patterns of ties to define and analyze the emergent structure. This is a significant benefit when there is not a "right answer" about what the structure should look like. "[N]etwork research has an *emancipatory potential* in that it can inform actors of non-obvious constraints and opportunities inherent in patterns of social connections" (Kilduff and Tsai 2003: 23). Finally, network analysis integrates quantitative, qualitative, and graphical data, thus allowing more thorough and in-depth analysis.

The basic element of NA is a *sociomatrix* that defines relationships between each row and column element. A one-mode matrix is built when the row and column elements are the same, as with frequency of interaction among a group of people or distance between cities. For symmetric one-mode data, the upper and lower halves of the matrix are identical. For directed one-mode data, the two halves are different. Directed data is collected on ties that are not reciprocal, as with authority structures or transfer of advice. For a matrix of product

development tasks collected in DSM format, directed data indicates the input / output flow of information between each task.

Table 1 contains a sociomatrix of directed data between nine nodes. Node A has unreciprocated ties with Nodes B and C, neither of which have ties to other nodes in the network. The empty upper half of this matrix shows that the ties in this network are directed, not reciprocal. The matrix in Table 1 would be symmetric if the ties were reciprocal. Network data are commonly displayed in graph form, where the nodes are the matrix row or column labels, and lines indicate the presence of a tie between two nodes. Figure 2 shows the network drawn from the data in Table 1. The arrows indicate a tie from the row actor to the column actor. For example, node A has arrows to nodes B, C, and D, while node B has arrows to nodes E and F.

Table 1: Example Sociomatrix

	Node A	Node B	Node C	Node D	Node E	Node F	Node G	Node H	Node I
Node A									
Node B	1								
Node C	1								
Node D	1								
Node E		1							
Node F		1							
Node G			1						
Node H				1					
Node I				1					

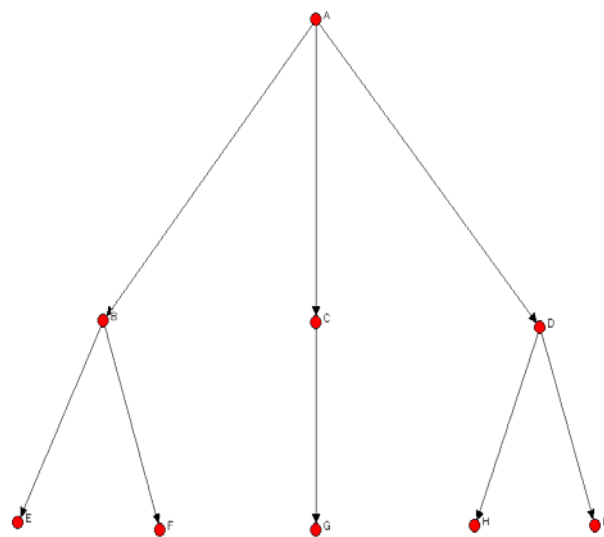


Figure 2: Network Drawn from Table 1

NA methods contain a wide variety of analytic techniques once the network data are collected in matrix form (Hanneman and Riddle 2005, Wasserman and Faust 1999). These methods target three different levels of analysis. First, it is possible to look at patterns of the overall structure (connectedness, density) and the location of individuals within the structure (centrality, geodesic distance). Second, it is possible to identify sub-structures (groups of nodes that are closer to each other than to other groups) based on patterns of relations. Examples are blocks, factions, cliques, and bridges. Third, it is possible to look at the positions of individuals within the structure. This is an analytical shift from viewing a node as having individually unique characteristics to viewing it as representing some categorical attribute. Demographic attributes are based on variable similarity like age (for people) or time to complete (for tasks). Network position analysis looks at attributes emerging from patterns of ties between nodes. For example, the “husband” role in a network is defined by a set of patterned interactions with categories like “wife” and “child.” Network categories like kinship are thus inherently relational roles rather than individual categories. Table 2 summarizes a set of key NA metrics for evaluating interacting product development tasks. The remaining sections of this paper discuss these metrics in more detail.

Table 2: Network Analysis Metrics for Evaluating PDP Task Interactions

SNA Term	Definition	Process Management Application
<b>Analysis Level: Overall Structure</b>		
Degree Centrality	Number of incoming and outgoing ties for each node	Critical process step based on frequency of interaction with other process steps
Betweenness Centrality	Number of times a node is on the shortest path between two other nodes	Critical process step based on quickest information transfer between two other steps
<b>Analysis Level: Sub-Structure</b>		
Clusters	Divide the network into N groups with maximized internal ties and minimized external ties	Identifies the N groups of iterative tasks that can be conducted concurrently
Density	Number of actual ties between nodes or groups divided by number of possible ties	Measures iteration within a group of concurrently scheduled tasks and between groups of ideally separate tasks
<b>Analysis Level: Individual Tasks</b>		
Influence	Degree of network dependence on a node (teacher of the next president)	Low visibility process steps that have potentially important information
Key Fragment Player	Node that will efficiently fragment the network if removed	Process steps that should not be omitted or done incorrectly
Key Reach Player	Node that will efficiently spread new information to the network	Process steps that are the best points to introduce new information or make design changes
Brokerage	What is the nature of B's role as a transmitter of information between A and C (co-ordinator, gatekeeper, representative, liaison, consultant)?	Characterizes role of key tasks within and between the N clusters

## Network Analysis of Product Development Tasks

This section discusses network analysis of Product Development tasks interactions at a small engineering company (Smallcomp). Smallcomp is a subsidiary of a multinational company (Multinat) that has a domestic and international presence in several industries. The first author (Collins) worked in the Engineering Management group from 2004 to 2006. This group is responsible for defining and maintaining Product and Technology Development processes at Smallcomp.

Smallcomp uses a common stage-gate system (Cooper and Kleinschmidt 1993, Cooper et al. 2002) for its product development activity as shown in Figure 6. The business gate reviews are held after each design phase to ensure that the maturing technical design still matches the expected market characteristics anticipated for the product deployment. This is initially done at Business Gate 0 to identify a market opportunity, and is done at Business Gate 4 to formally release the product for sales. Each business gate is preceded by an engineering design review to evaluate the product's technical feasibility and risk relative to the defined requirements. The Concept Review is held before Business Gate 1, System and Powerplant Design Reviews are held before Business Gate 2, Critical Design Review is held before Business Gate 3, and Validation Review is held before Business Gate 4.

Smallcomp began investigating matrix-based project management tools during a quality improvement initiative to streamline its product development task execution. This resulted in a set of task interactions being captured and evaluated using a traditional time-based DSM. The remainder of this section uses that DSM data to discuss how matrix analysis using NA diagnostics can provide insight that guides process management activity. The diagnostics applied to this data are summarized in Table 2.

## Overall PDP Structure

At this level of analysis, NA metrics treat the product development process as a single entity of information-sharing tasks. Table 2 identifies two key metrics. Degree Centrality measures the number of incoming and outgoing ties that each task has. Betweenness Centrality measures the number of times a particular task is on the shortest path between two other tasks. The steps with the highest Betweenness Centrality are Design Point Definition, Detailed Heat and Mass Balance, P&ID and Electrical 1-line, Business Gate 1, and Audit Preliminary Design. The PDP network is very dependent on these because of their high “betweenness centrality” value, analogous to the critical path identification in MS Project.

Figure 3 shows control charts for degree centrality based on the number of incoming ties (InDegree) and outgoing ties (OutDegree). Relative to the other tasks in the process, the tasks above the limit line have higher than average role to coordinate either incoming or outgoing information. This is one metric of task criticality or workload in the network. The workload on tasks in this product development process is to receive input from between four and ten other tasks, and provide output to between two and eleven other tasks. Figure 3 rapidly summarizes the outlier tasks that have relatively high or low information transfer loads. The tasks with high InDegree centrality, meaning they require more than the normal input, are the customer specification review (P100), design and development plan (P150), concept design review (P165), Business Gate 1 (P195), and test stand preparation (P253). The tasks with low InDegree centrality, meaning they require less than the normal input, are business gate 0 (P20), reviewing lessons learned (P105), establishing design approvals and charge numbers for preliminary design (P205). The tasks with high OutDegree centrality, meaning they provide more than the normal output, are Business Gate 0 (P20), establishing IPD teams for the concept phase (P103), defining process, electrical, and mechanical integration topologies (P120A, P120C, and P120E), auditing preliminary design trade studies (P215D), and verification testing (P420). The tasks with low OutDegree centrality, meaning they provide less than the normal output, are reviewing lessons learned (P105), concept review (P165), establishing design approvals and charge numbers for preliminary design (P205), verifying controller software (P360), procuring experimental hardware (P450), and Business Gate 4 (P495).

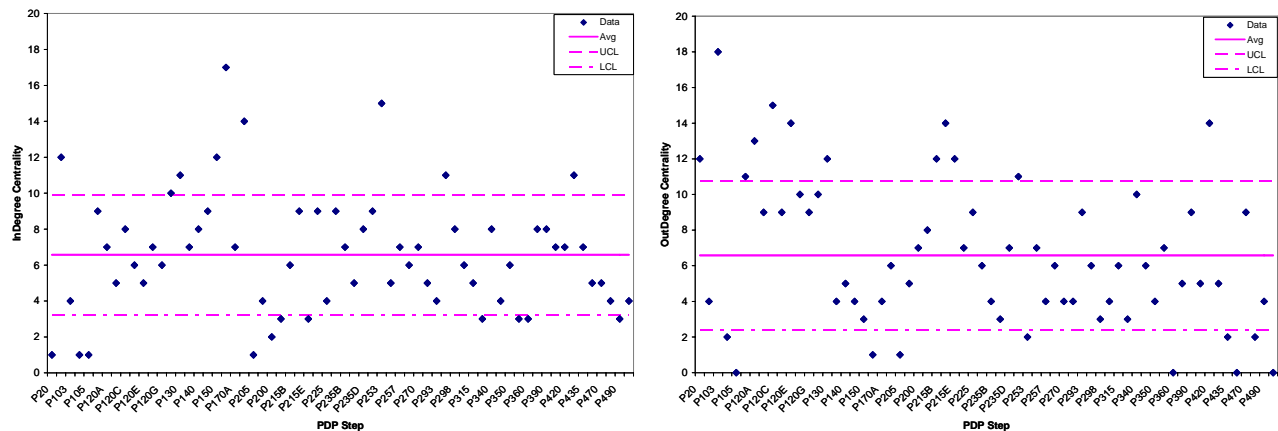


Figure 3: Control Chart for Degree Centrality

### PDP Sub-Structures

At this level of analysis, NA metrics identify and measure groups of tasks that interact more with each other than with other tasks in the product development process. Table 2 identifies two key metrics. Clustering identifies groups of tasks that are strongly connected to each other as a basis for scheduling concurrent activity. Density measures the number of actual and potential ties within a cluster, and between clusters. This describes the strength of concurrency, as well as the strength of coupling between groups of tasks.

Table 3 clusters five groups of tasks to maximize within-group interaction and minimize between-group interaction. The diagonal values in Table 3 measures the density of ties within the cluster. The off-diagonal measures the density of ties between clusters. For example, there are 41% of the potential connections within the System Definition cluster, 3% of the potential connections from System Definition outputs to Preliminary Functional Integration inputs, and 1% of potential connections from Preliminary Functional Integration outputs to System Definition inputs. Several tasks conducted early in PDP execution (P20 – Business Gate 0, P100 – Customer Specification Review, Set up Design File, and P105 – Lessons Learned) are placed in the Validation cluster based on loose ties to the rest of the PDP network. This placement shows that the relationship of these tasks to other System Definition tasks was not well defined in the original DSM.

Table 3: Clusters of Concurrent PDP Tasks

	System Definition (Circle)	Preliminary Functional Integration (Triangle)	Preliminary Mechanical Integration (Crossed Square)	Detailed Design & Verification (Inverse Triangle)	Validation & Other (Square)
System Definition	<b>0.41</b>	0.03	0.02	0.00	0.06
Preliminary Functional Integration	0.01	<b>0.39</b>	0.18	0.05	0.03
Preliminary Mechanical Integration	0.00	0.11	<b>0.37</b>	0.05	0.00
Detailed Design & Verification	0.00	0.02	0.00	<b>0.45</b>	0.10
Validation & Other	0.08	0.02	0.00	0.03	<b>0.16</b>

The low off-diagonal density values in Table 3 show that the clustering intent, which was to identify groups of iterative or concurrent tasks that are independent of the rest of the PDP activities, was mostly successful. The primary exception is the ties between the Preliminary Functional and Mechanical Integration clusters. The Functional – Mechanical tie is relatively strong (18% between-group compared to 39% and 37% within-group), while the Mechanical – Functional tie is moderate (11% between-group compared to 39% and 37% within-group). This shows that while functional and mechanical integration can be defined as separate sets of activities, there are interdependencies between them that must be actively managed.

## Individual PDP Tasks

At this level of analysis, NA metrics identify key tasks based on their relational roles as information transmitters in the product development process network. Table 2 identifies four key metrics. Key Player analysis (Borgatti 2003) identifies tasks that are efficient points for introducing new design information, or whose omission adds risk to PDP execution. Although this is conceptually similar to centrality, there are mathematical and practical alternatives to choosing the set of nodes with highest betweenness centrality to fragment a network, or highest closeness centrality to diffuse the network. The key player algorithm uses combinatorial optimization to find the best combination of nodes to remove or use as diffusion initiators.

Table 4: PDP Key Player “Fragment” Steps

PDP Step
P195: Business Gate 1 – Released Requirements and approval to proceed with preliminary design.
P200: Establish Organization For Design & Development Phase
P215B: P&I Diagram & Electrical 1-line
P215D: Audit Preliminary Design – Select "Best" Alternative
P215E: Detailed Heat & Mass Balance
P230: Prelim PP Package - Assembly drawings, BOM, ICD

The six “Fragment” nodes in Table 4 lists steps that, if removed, would split the PDP network into multiple, less well-connected task groups. Omitting or completing these tasks incorrectly contributes to fragmented information in the remaining elements of the project. These steps are opportunities for management “containment intervention” to ensure they are not overlooked or inadequately completed. In general, each of these tasks was already recognized as important for Smallcomp’s project execution. For example, many people identified accurate organizational design as an issue during several “churn elimination” brainstorming sessions in 2005. This feedback was a key element of an internal procedure revision to provide guidelines that IPD teams not be formed until after the System PDR.

The ten “Reach” nodes in Table 5 are PDP steps that are the most efficient points of information diffusion. These tasks are opportunities for management “proactive intervention,” meaning they are the most effective points to hold discussions that may result in changes to design content or project scope. These steps should be highlighted during project planning as the best points at which to introduce new information or design changes. 70% of the PDP network can be reached using Business Gate 0, Concept Design Review, and Business Gate 1. New information introduced after Business Gate 1 will not reach all of the remaining tasks unless the PDP execution is changed.

These results highlight the importance of managing information at the PDP interface when programs are beginning. Two particular areas should be noted. First, Technology Development tasks could be key players for diffusing information through the PDP network via insertion of advanced technology from Smallcomp’s research activities. Second, some Product

Management tasks could be key points to diffuse information through the PDP network via insertion of market or customer information into the Engineering activities in this network.

Table 5: PDP Key Player “Reach” Steps

PDP Step	Nodes Reached	% of PDP Network
P20: Business Gate 0: Approval to develop concept design	21	32.8
P165: Conduct Concept Design Review - Approval to proceed with system design.	36	56.3
P195: Business Gate 1 - Released Requirements and approval to proceed with preliminary design.	44	68.8
P215D: Audit Preliminary Design - Select "Best" Alternative	50	78.1
P257: Controller Software Requirements	55	85.9
P295: PDR for Power Plant, Assembly, & Components	59	92.2
P340: Detailed Power Plant Design (Detailed Assembly Drawing)	61	95.3
P420: Define and Conduct Verification tests	62	96.9
P425: Business Gate 3 - Approval to proceed with Verification & Validation Phase.	63	98.4
P475: Prepare installation and operation manuals	64	100.0

Another metric of individual position is *brokerage*. Brokerage occurs for the triad of nodes A, B and C in Figure 4. In this network, A has a tie to B, B has a tie to C, but A has no tie to C. B is a broker because A needs B to reach C. Expected brokerage calculates the number of times each node in a cluster would be expected to play each brokerage role based on the number of clusters and size of each cluster. Relative brokerage ratios the actual number of roles each node plays against the expected number of roles. High relative brokerage values indicate a node is playing a particular role more than expected. Low relative brokerage indicates the node is playing a role less than expected.

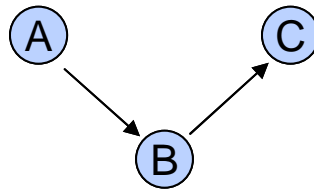


Figure 4: Example Brokerage by Node B

Node B has five potential brokerage roles between nodes A and C. B is a *coordinator* if all three nodes are in the same group. B is a *consultant* if it is in a different group from A and C. B is a *gatekeeper* if A is in a different group from B and C. B is a *representative* if C is in a different group from A and B. B is a *liaison* if all three nodes are in different groups. Brokerage identifies tasks that have critical information-sharing roles within and between the clusters in Table 3.

Many programs will modify the baseline PDP based on external constraints such as compressed schedule, reduced resources, or altered scope. This increases the burden of coordinating information transfer between program participants. Figure 5 shows the number of tasks in each PDP cluster where the relative brokerage value is greater than 3. Relative brokerage ratios the actual number of ties to the expected number of ties based on the cluster size. The magnitude of the ratio describes the burden for the particular role that the associated

task must fulfill. Brokerage is a useful way to examine how information transfer is taking place within and between groups of PDP tasks. High coordination activity identifies tasks that are critical for within-group execution. The other four roles identify tasks that are critical for between-group execution. Identifying these roles adds another dimension to interface management between groups of tasks, and between the assemblies or teams executing those tasks. This is particularly valuable when tasks are being completed by distributed teams. Identifying tasks with high brokerage roles is important for defining and coordinating information transfer between these teams.

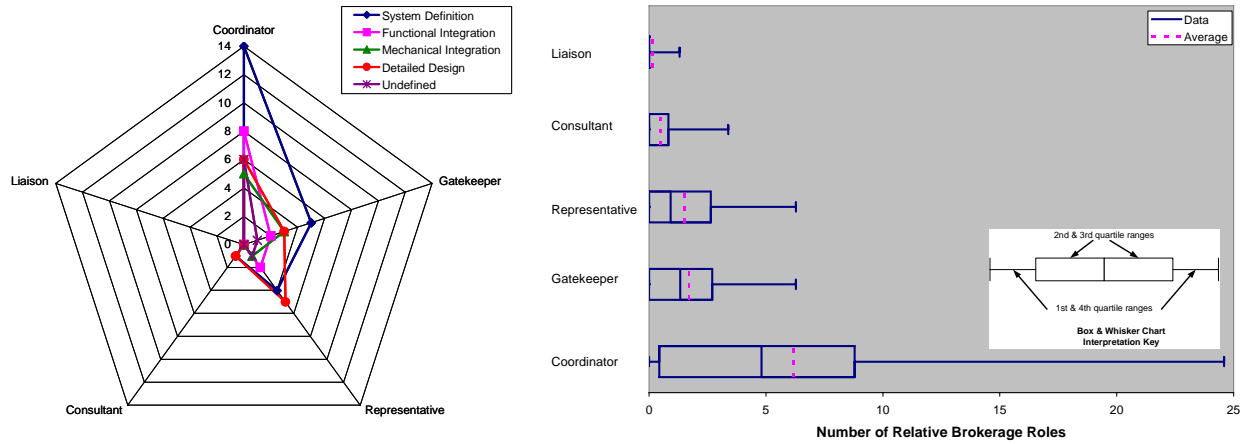


Figure 5: Summary of Task Brokerage Roles

The type of coordination may vary across different clusters of PDP tasks. For example, all five task clusters in Figure 5 contain several tasks with higher than expected coordination roles. This attention to the required coordination *within* a task cluster is expected based on Smallcomp’s desire to define clusters of concurrently scheduled or iterative activities. The high Gatekeeper and Representative values for the System Definition and Detailed Design clusters indicate tasks that are important for coordinating information transfer *between* task clusters. Coordination burden decreases as the process progresses from System Definition to Validation.

The highest number of coordination brokers is in the System Definition phase, followed by Functional Integration, then Mechanical Integration and Detailed Design. System Definition and Detailed Design both have tasks with high relative consulting brokerage. This means that two tasks from the same cluster must pass information to each other through a task in a different cluster. The consultant task for System Definition is P135. The consultant task for detailed design is P390. System Definition has a higher number of tasks acting as gatekeepers, again indicating fewer tasks controlling information flow to another cluster as the process progresses. Detailed Design has a high number of representative tasks that take information from another cluster and provide to internal tasks. This is again consistent with the idea of funneled information flow. The interesting finding here is that System Definition has many representative tasks as well. These are the tasks that receive feedback from later activities that impact revisiting the concept activities.

## Discussion

### Visualization Tools for Process Management

Figure 6 shows a network map of the entire Smallcomp PDP based on the DSM data. The node color identifies the key players defined in Table 6. The node shape is based on the five task “clusters” in Table 3. The task execution in Figure 6 moves from right (early program activity) to left (field testing and support). However, the order is clearly not a linear sequence. Traditional process management techniques such as value stream mapping which rely on visual process description to identify key process characteristics are difficult to apply in this case due to the dense interconnections between tasks.

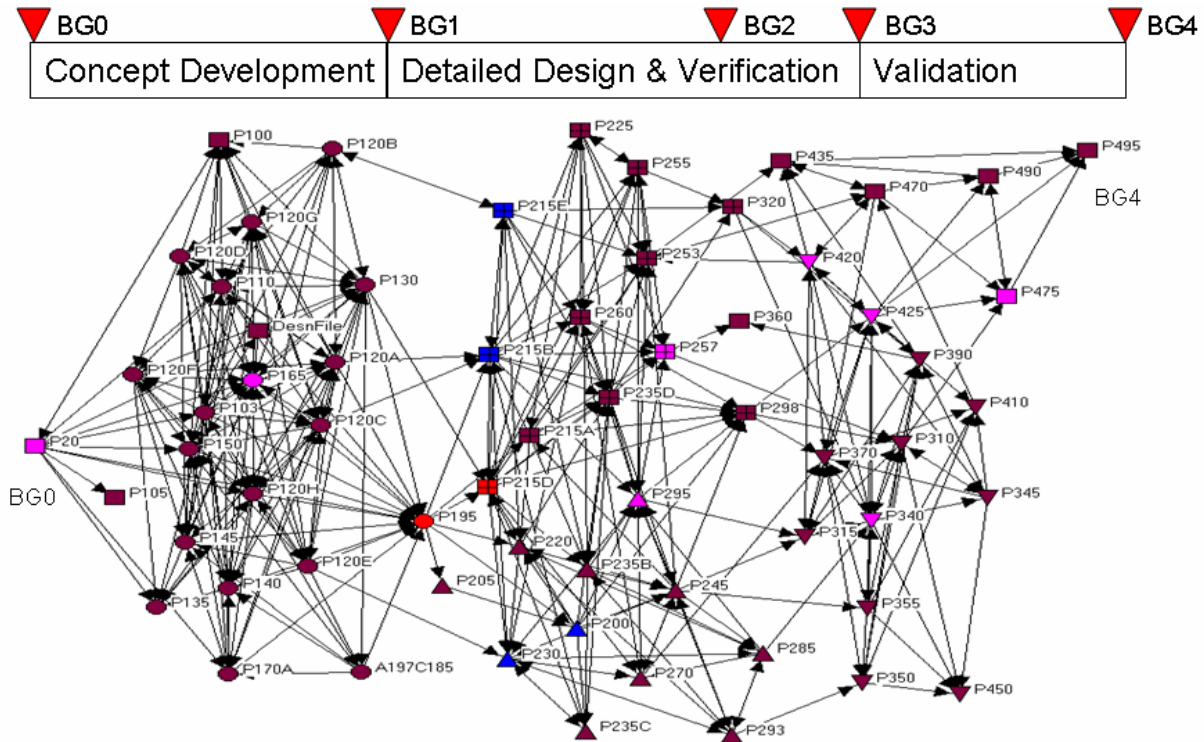


Figure 6: Network Map of Smallcomp's Product Development Tasks

Table 6: Key Player Tasks in Figure 6

Group	# of Nodes	Node Color	Node Graph ID
Reach & Fragment	2	Red	P195, P215D
Fragment only	4	Blue	P200, P215B, P215E, P230
Reach only	8	Pink	P20, P165, P257, P295, P340, P420, P425, P475
Non-key	50	Brown	All others

Graph theory methods for visualizing networks are a powerful way to display information within a DSM. For example, overlaying the business gate process with the task interactions in Figure 6 clearly shows that the business gates are not acting as the checkpoints they are supposed

to. One external reviewer of the paper, on seeing this figure, commented that “You probably do not have a way to control projects that are in between Gate 1 and Gate 4, and you probably have significant cyclical activity between your design, verification, and validation phases.” This is a remarkably accurate assessment, considering it was made by someone with no knowledge of Smallcomp’s recent product development history. It is not something that would be immediately obvious from the business gate flow chart alone, nor would it be clear from a classical DSM matrix display.

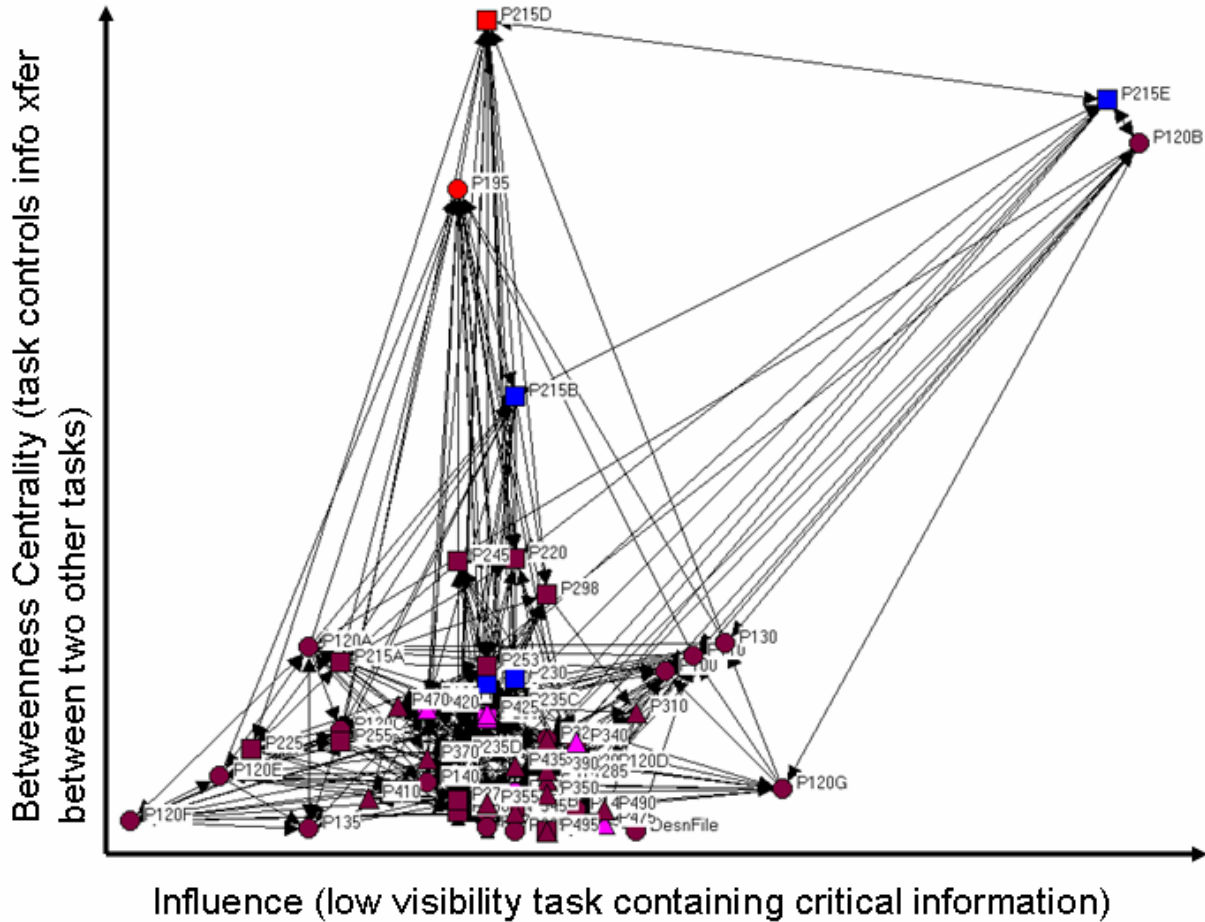


Figure 7: PDP Tasks by Influence & Betweenness Centrality

Figure 7 shows a re-drawn PDP network map where x-axis co-ordinate places the nodes by increasing influence and the y-axis co-ordinate places the nodes by increasing betweenness centrality. It visually summarizes important tasks based on characteristics that are important to PDP execution (in this case centrality and influence). The most influential steps are P215E (Detailed Heat and Mass Balance) and P120B (Design Point Definition). The least influential steps are P120F (Define Power Plant Package) and P120E (Identify Major Assemblies). The steps with highest betweenness centrality, meaning they are most frequently on the shortest path between two other steps are P120B (Design Point Definition), P215E (Detailed Heat and Mass Balance), P215B (P&ID and Electrical 1-line), P195 (Business Gate 1), and P215D (Audit Preliminary Design). Figure 7 condenses interactions between all the PDP tasks to show the six or seven critical tasks. It makes intuitive sense that the steady state model and design points

would be highly influential, because these are tasks that other tasks need information from. The uniqueness of these tasks would not be immediately obvious using a swimlane, value stream map, or even normal network map. These tasks would be hard to identify in Figure 6.

### ***Catalyzing Heuristic Discussions (Tradeoffs)***

Filtering a small list of tasks from the entire PDP enables more effective discussions targeted at process management. Figure 7 was a valuable starting point for discussions with a group of Smallcomp's Engineering managers. It validated earlier comments about churn that results from poor design point definition, and raised the need to be more intentional about assembly drawings. Attributing low influence to P120E and P120F was consistent with legacy programs where packaging was not a major design constraint. They suggested the two steps be re-examined since packaging is a major design constraint for current Smallcomp programs.

Charts like this could also be used for another set of discussions: Are these tasks as important as the diagnostics claim? If so, do they get appropriate attention during program execution? If not, what changes need to be made on future programs? Where are the steps we claim are critical, like technical and business reviews? Should they be showing up with high values on key analytical dimensions (in this case influence and centrality)? If they are not, then why not?

### ***Lessons Learned***

Bottoms-up process definition doesn't immediately result in well coordinated interaction patterns. The interesting thing here is the potentially discrepant definitions from the same group of experts. The business gate system was put in place by a group of experienced engineers tasked with identifying a development process Smallcomp could use that would be consistent with other Multinat divisions, as well as represent industry best practices. When the original DSM work was done, the same group of experts were interviewed to identify individual task interaction patterns. The idea of process / practice discrepancies, or conflicting schemas is not a surprising finding. Expect one answer when you ask someone what they think *should* be happening, and expect a different answer when you ask someone what they think *actually* happens. The danger is that the DSM task list came out of a request that the experts identify a checklist of guidelines that development programs could use to help plan their activities. This is actually a case of process / process discrepancy, not process / practice discrepancy.

For brokerage, visual display can show that the business gate steps are not acting as the check points they're envisioned to be. There are two alternatives here. First, redesign the process to force everything through the business gate steps. Second, use brokerage analysis to look at the tasks that ARE acting as gatekeepers between passport phases (since the passports are note), and focus energy on making sure those steps are robust. This has to be an iterative process, but use the relative brokerage values as a starting point. For example, Figure 5 condenses the brokerage roles using a relative brokerage value of 3.

## **Conclusion**

A significant element of engineering program execution is transmission of information between groups. NA techniques such as key players, centrality, influence, brokerage, and factions provide a method to identify critical product development tasks and interactions as part of ongoing process improvement activity. Managers seeking to make meaningful improvements

in the product development cycle may often incorrectly identify actual sources of interdependencies because of the complex and interacting nature of organizational processes, tasks, and relationships (Siggelkow 2002). The NA metrics described in this paper provide a way to analytically determine the key relationships in complex product development environments, rather than relying on an individual or set of individuals to accurately understand the entire process. This allows product development processes to be emergent, rather than pre-defined phenomena. One benefit is the ability to describe feedback characteristics of critical product development tasks. This is especially valuable when evaluating the impact of omitting, combining, or re-sequencing task execution based on program-specific constraints.

## References

- Batallas, D. and Yassine, A., *Information Leaders in Product Development Organizational Networks: Social Network Analysis of the Design Structure Matrix*. IEEE Transactions on Engineering Management, 43(4), 2006. pp 570-582.
- Borgatti, S. P., *The Key Player Problem*. R. Breiger, K. Carley and P. Pattison (Eds), *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*. National Academy of Sciences Press, 2003. Pp 241-252.
- Browning, T. R., *Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions*. IEEE Transactions on Engineering Management 48(3), 2001. pp 292-306.
- Browning, T., Fricke, F. and Negele, H., Key Concepts in Modeling Product Development Processes. Systems Engineering 9(2), 2006. pp. 104-128.
- Bustnay, T. and Ben-Asher, J., *How many Systems Are There? – Using the  $N^2$  Method for Systems Partitioning*. Systems Engineering 8(2), 2005. pp. 109-118.
- Cooper, R. G. and Kleinschmidt, E. J., *Stage Gate Systems for New Product Success*. Marketing Management 1(4), 1993. pp 10-29.
- Cooper, R. G., Edgett, S. J. and Kleinschmidt, E. J., *Optimizing the Stage-Gate Process: What Best Practice Companies Do - II*. Research Technology Management 45(5), 2002. Pp 21-27.
- Hanneman, R. A. and Riddle, M., *Introduction to Social Network Methods*. University of California, Riverside (published in digital form at <http://faculty.ucr.edu/~hanneman/>), Riverside, CA, 2005.
- Harris, T., Grover, R. and Hariharan, N., *Roadmapping Smallcomp Design Process*. Capstone Project: MIT Certificate Course in Systems Engineering, 2003.
- INCOSE, *Systems engineering handbook: A “what to” guide for all SE practitioners*, International Council on Systems Engineering, 2004.
- Kilduff, M. and Tsai, W., *Social Networks and Organizations*. Sage Publications, Thousand Oaks, CA, 2003.
- McCord, K. and Eppinger, S. D., *Managing the Integration Problem in Concurrent Engineering*. MIT Sloan School of Management Working Paper no. 3594, 1993.
- Scott, J., *Social Network Analysis: A Handbook*. Sage Publications, Ltd., London, 2000.
- Seshadri, P. and Thompson, V., *Analysis and Optimization of Concurrent Product and Technology Development Programs for the Development of Fuel Cell Systems*. Capstone Project: MIT Certificate Course in Systems Engineering, 2006.
- Sharman, D. and Yassine, A. “Characterizing Complex Product Architectures,” Systems Engineering Journal, Vol. 7, No. 1, 2004.

- Siggelkow, N., *Misperceiving Interactions among Complements and Substitutes: Organizational Consequences*. Management Science 48(7), 2002. Pp 900-917.
- Sosa, M. E., Eppinger, S. D. and Rowles, C. M., *Identifying Modular and Integrative Systems and Their Impact on Design Team Interactions*. ASME Journal of Mechanical Design 125(June), 2003. Pp 240-252.
- Sosa, M. E., Eppinger, S. D. and Rowles, C. M., *The Misalignment of Product Architecture and Organizational Structure in Complex Product Development*. Management Science 50(12), 2004. Pp 1674-1689.
- Tapping, D. and Shuker, T., *Value Stream Management for the Lean Office*. Productivity Press, New York, NY, 2003.
- Wasserman, S. and Faust, K., *Social Network Analysis Methods and Applications*. Cambridge University Press, Cambridge, 1999.
- Yassine, A., Joglekar, N., Braha, D., Eppinger, S. and Whitney, D., *Information Hiding in Product Development: The Design Churn Effect*. Research in Engineering Design 14(3), 2003. Pp 145-161.
- Yassine, A. and Braha, D. *Four Complex Problems in Concurrent Engineering and the Design Structure Matrix Method*. Concurrent Engineering Research & Applications 11(3), 2003.
- Yu, T-L., Yassine, A. and Goldberg, D. "A Genetic Algorithm for Developing Modular Product Architectures," Proceedings of the ASME 2003 International Design Engineering Technical Conferences, 15th International Conference on Design Theory & Methodology, Sept. 2-6, 2003. Chicago, Illinois.

### Author Information

**Shawn Collins** is a PhD candidate in Anthropology at the University of Connecticut. His research interests are in sociotechnical systems engineering, organizational behavior, and business anthropology. He received the B.S. degree in Mechanical Engineering from Purdue University in 1999. He received the M. A. degree in Anthropology in 2003 from the University of Connecticut and the M. S. degree in Mechanical Engineering in 2004 from Rensselaer Polytechnic Institute. He is a member of the Society for Applied Anthropology and the American Anthropological Association. He has worked as a systems engineer in the aerospace and energy industries.

**Ali Yassine** is an assistant professor in the Department of Industrial and Enterprise Systems Engineering (*IESE*) at the University of Illinois at Urbana-Champaign (UIUC) and the director of the Product Development Research Laboratory. His research involves managing the development process of complex engineering systems, design process modeling, and IT-enabled concurrent engineering. Dr. Yassine received the B.E. degree in Mechanical Engineering in 1988 from the American University of Beirut. He received the M.S. and Ph.D. degrees in 1989 and 1994 in Industrial and Manufacturing Engineering from Wayne State University in Detroit, Michigan. He is a member of INFORMS, ASME and PDMA.