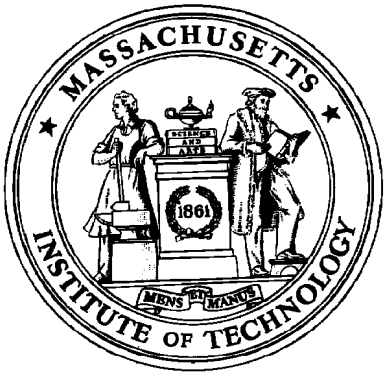

The Lean Aircraft Initiative

Report Series

#RP97-01-19

February 1997

Summary Report: Systematic IPT Integration in Lean Development Programs

Prepared by: Tyson Rodgers Browning

Lean Aircraft Initiative
Center for Technology, Policy, and Industrial Development
Massachusetts Institute of Technology
Massachusetts Avenue • Room 33-407
Cambridge, MA 02139

The authors acknowledge the financial support for this research made available by the Lean Aircraft Initiative at MIT sponsored jointly by the US Air Force and a consortium of aerospace companies. All facts, statements, opinions, and conclusions expressed herein are solely those of the authors and do not in any way reflect those of the Lean Aircraft Initiative, the US Air Force, the sponsoring companies and organizations (individually or as a group), or MIT. The latter are absolved from any remaining errors or shortcomings for which the authors take full responsibility.

This document provides a summary report of the M.I.T. masters thesis _Systematic IPT Integration in Lean Development Programs_ by Tyson R. Browning (complete reference at end of document).

Outline of Contents

1. Overview

Integrated product development (IPD) has been established as a salient aspect of the lean enterprise paradigm. The drive towards IPD includes an impetus to organize around *integrated product teams* (IPTs), cross-functional groups responsible for developing a particular element of a system product. The use of IPTs has brought with it many issues, including those at the IPT interfaces. The goal of this research is to provide insight into the integration of multiple IPTs in programs. (The research is sensitive to the fact that the defense aircraft industry is unique, differing in significant ways even from its close kin, the commercial aircraft and automotive industries. Nevertheless, the latter is the launching point for several key ideas that potentially map into the defense aircraft industry.)

Program integration of a cross-functional, upstream/downstream nature is required at three distinct realms or levels: (1) within IPTs, (2) between IPTs but within higher level team groupings, and (3) between IPTs and higher level team groupings in a program at large. This work focuses on the second and third levels, realms of IPT interdependence, and categorizes several *integrative mechanisms* (IMs) that facilitate interteam integration. IMs are strategies and tools for effectively coordinating actions across team interfaces.

To provide a variety of contexts for investigation, the research includes five case studies of development programs of varying characteristics. Each is analyzed for its use of IMs. The five programs are: (1) GEN–X Program, Texas Instruments (defense aircraft electronics), (2) F/A–18E/F Program, McDonnell Douglas (defense aircraft airframes), (3) Turbine Airfoils Center of Excellence and F110+ Programs, General Electric Aircraft Engines (defense aircraft engines), (4) 737–700 EPGS (IDG) Program, Sundstrand (commercial aircraft), and (5) Small Car Platform and

Neon Program, Chrysler (commercial automobile). Non-defense aircraft programs are included for the sake of comparison and contrast.

The application of a people-based *design structure matrix* (DSM) to model program organization based on information flow is considered in one case. DSM approaches to interface management have been used in the automotive and other industries. The people-based DSM enables a systematic approach to the interface management process at an interteam level. This facilitates decisions as to the appropriate utilization of IMs. To assist in implementing a systematic approach to program organization and integration, a framework process, *design for integration* (DFI), is presented.

These studies argue for the inclusion of program integration principles as an essential aspect of lean enterprise product development and organization.

2. The Necessity of Multi-team Integration

While the thesis discusses in more detail the background and motivation for IPD, concurrent engineering, and IPTs, these concepts are taken as given for the purposes of this summary. An important characteristic of IPTs worth mentioning here, however, is that they are ideally on the order of ten people[1] in size and are usually chartered to develop a particular element of an overall system—i.e., they are the lowest level cross-functional team.

The design of complex system products involves hundreds or thousands of individuals making millions of design decisions over months or years. Because of couplings within the system architecture, these decisions require personal interaction among the designers: design choices involve tradeoffs which affect many other design, process, cost, risk, manufacturing, and operational parameters. Managers' "primary development challenge is to integrate the many sub-problem solutions into a well-designed system. … The trouble is that such interactions are often poorly understood and are rarely known in advance."[2]

The key to managing the organizational interfaces effectively lies in ensuring the proper interactions between many IPTs and functional groups. "Integration of teams' activities can … be a nightmare. Like functionally oriented organizations, teams can get tunnel vision and forget that there are other missions and reasons for an organization's existence."[3] Not only must the product teams be cross-functionally integrated as IPTs, but also the IPTs themselves must be integrated on a higher level. Therefore, there exists at least a second level of integration within each program. The framework presented in §3 captures these distinct levels.

---

[1] In a range of about three to twenty-five people
[2] (Eppinger 1994), p. 1
[3] (Cole 1995), p. 30

Large, complex system development programs experience great difficulties integrating all functions into small IPTs. Often, this is not practical or even feasible. Various *functional support groups* (FSGs) such as test labs remain outside IPTs' jurisdictions. Such situations challenge an IPT approach to IPD. Yet perhaps certain IMs can also successfully integrate these groups at a higher level in the program.

The program design process becomes increasingly difficult as system complexity increases. (Complexity here implies numerous, highly-coupled subsystems and elements.) The IPTs working together to develop such a system face a daunting task. Team A needs to know what values team B has set for parameters *x* and *y*; team B needs to know what values team C is using for parameters *w* and *z*; but team C needs to know the result of team A's activities to determine *w* and *z*. Coupled "chicken and egg" problems may imply a slow, iterative development process. Without precaution, the number of required interfaces and thus the number of iterations required to converge to an acceptable design can increase exponentially with system complexity. As the amount of task and IPT interdependence increases, the traffic on inter-IPT communication channels increases. The results of a study by Tausworthe add the following conclusion: "A team producing at the fastest rate humanly possible spends half its time coordinating and interfacing."[4]

As Rechtin notes, "The greatest leverage in system architecting is at the interfaces."[5] While true for a system architecture, it also applies to the IPTs that develop a system's constituent parts. A better understanding of the issues underlying IPT interdependence would seem to hold great potential for improving the product development process.

3. A Program Integration Framework

It is convenient to think of integration as necessary at three levels within a program. McCord and Eppinger have drawn a distinction between IPTs and "system teams"—sets of IPTs and FSGs pulled together because of mutual, high traffic interfaces.[6] IPTs represent a first level of integration; system teams are a second. Adding the overall program as another level, the proposed framework then consists of three levels: IPT level (first level), system level (second level), and program level (third level). Figure 1 provides an illustration. Individual IPTs (and FSGs, if applicable) can be aggregated in system level teams, which, along with other IPTs (and FSGs) and system teams, comprise a program.

---

[4] Qtd. in (Rechtin 1991), p. 284

[5] (Rechtin 1991), p. 29

[6] (McCord 1993) Technically, it is probably more correct to refer to these groupings as "subsystem teams," since the organizational entity formed often consists of a group of IPTs working on the respective elements of a particular subsystem—one of many in the overall system which the program as a whole is developing.

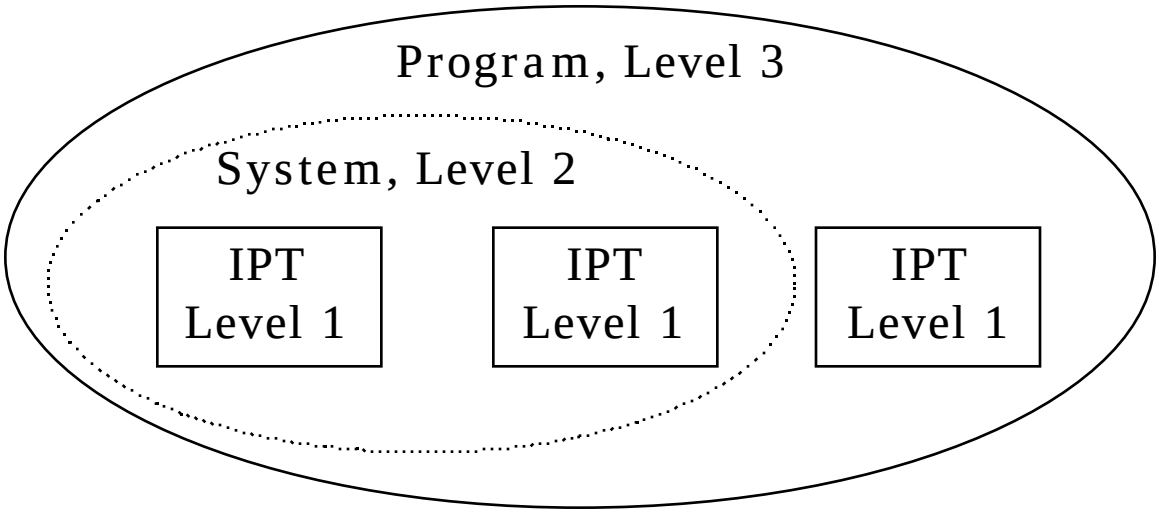

Figure 1: Three Levels of Integration

Integrating at the various levels reveals tradeoffs. For example, should a given functional representative be made available to and integrated within a particular IPT? or will some resources have to be shared by many IPTs and instead integrated at level two or three? The addressing of these tradeoffs and the appropriate application of IMs to facilitate inter-IPT and system team integration are the subject of the proposed DFI process.

The three level framework is similar to multiple level organizations used by the F/A-18E/F program and other aerospace projects. In some cases, these programs expand to four or five levels to capture finer distinctions between system team sizes and/or priorities. Thus, the three level model represents a minimum amount of distinction between levels for a large, complex system development project.

4. Integrative Mechanisms (IMs)

Integrative mechanisms (IMs) are strategies and tools for effectively coordinating actions between IPTs and FSGs. IMs are catalysts, facilitating information flow across organizational, locational, cultural, traditional, and other barriers. They must regulate information flow such that it does not overwhelm or "underwhelm" its recipients. The categories of IMs explored in this research and listed below represent the tools in an integrator's "tool kit."

1. **Systems engineering and interface optimization**—designing the organization to mirror the system product architecture. An intelligently decomposed and partitioned architecture (such that interfaces between subsystems are minimized) will facilitate interteam interface management.
2. **Improved information and communication technologies**—linked CAD/CAM/CAE systems, e-mail, tele- and videoconferencing, common databases (easily accessed and shared), common nomenclature, etc.

3. **Training**—especially in team-building (and "system team-building" and "program-building"); raising awareness about integration needs and roles
4. **Co-location**—physical adjacency of IPT, FSG, system team, and/or program members
5. **"Town meetings"**—not to share technical information, but to boost camaraderie and to increase awareness of program-wide issues
6. **Manager mediation**—"up-over-down" (hierarchical) issue mediation schemes; *heavyweight product managers* (HPMs) or *integrators*
7. **Participant mediation**—*liaisons, engineering liaisons, conflict resolution engineers*
8. **Interface "management" groups**—integration teams tasked with ensuring ongoing or incident-specific mediation of interface issues
9. **Interface contracts and scorecards**—explicit delineation of interface characteristics and metrics for evaluating interface effectiveness

5. The People-Based Design Structure Matrix (DSM)

5.1 What is a People-Based DSM?

A people-based DSM is a square matrix where row *i* and column *i* both represent team or group *i* out of a list of IPTs and FSGs. A people-based DSM shows the interactions between groups—specifically, of each IPT and FSG with every other IPT and FSG. By reading across a row, one can see which other teams that row's team interacts with. A DSM is able to capture a large number of interfaces in a relatively simple format. Complex relationships between teams are visible. A DSM is especially useful for identifying clusters of activity and tracking obscure interfaces. With minimal training, a group of managers can discuss integration issues from a common, highly visual framework. A DSM-based modeling approach enables IM #1 (systems engineering and interface optimization).

A people-based DSM will have IPTs and FSGs listed down the left side (one per row) with an associated letter or number. Figure 2 shows an example for the development of a new soda bottle, where a team is assigned to each of the key tasks listed. An "**x**" in the matrix signifies an information flow between teams. (Diagonal elements are just placeholders, since a team does not interact with itself at this level.) Reading across a row reveals what other teams the team in that row *depends* on for information, and reading down a column reveals what other teams the team in that column *provides* information to.[7] For example, team B depends on information from teams A, C, D, F, H, and I, while team C must provide information to teams B, E, F, H, and I. The DSM renders a snapshot of where information flows in the project.

[7] Reading across a row reveals the sources of inputs to a task, while reading down a column indicates where the outputs of a task must go, the "sinks".

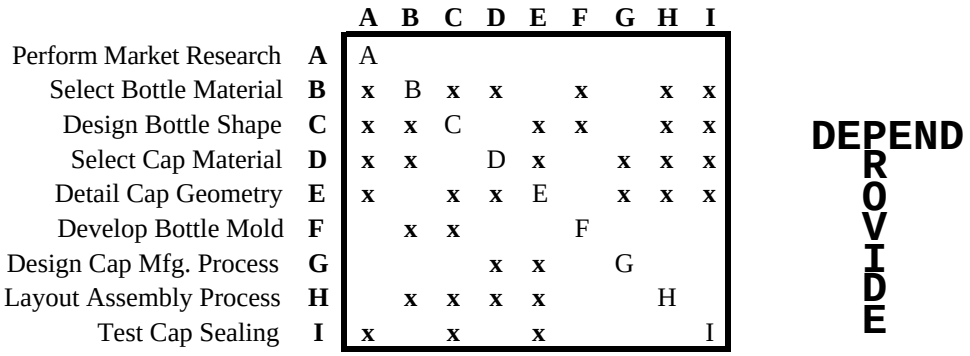

Figure 2: People-Based DSM Example—Development of a New Soda Bottle[8]

5.2 The People-Based DSM Facilitates Systematic IPT Integration

Complex system development requires the exchange of technical information among groups. Understanding the sources and sinks of this information flow can provide insights leading to improved organization and integration. It is important that decisions affecting entire programs and organizations proceed with a systematic foundation. DSM analysis enables a more systematic approach and informs tradeoffs regarding the integration of IPTs and FSGs at appropriate organizational levels and in the application of IMs.

The approach involves clustering IPTs and FSGs into groups with high internal integration needs and minimal external interfaces. Note that this is the same, general systems approach and objective recommended for product architectures themselves. (That is why a wise product decomposition and an organization based upon the product architecture greatly facilitates organizational integration.) For example, consider the frequency and direction of information flow data in an automobile engine development program, as shown in Figure 3.[9] This DSM is a three-dimensional array: the level of information dependence can be "low," "average," or "high." Note the extent of the interdependencies among the product development teams (PDTs).

[8] Source: (McCord 1993), p. 8
[9] This example is taken from (McCord 1993), where it is developed more extensively.

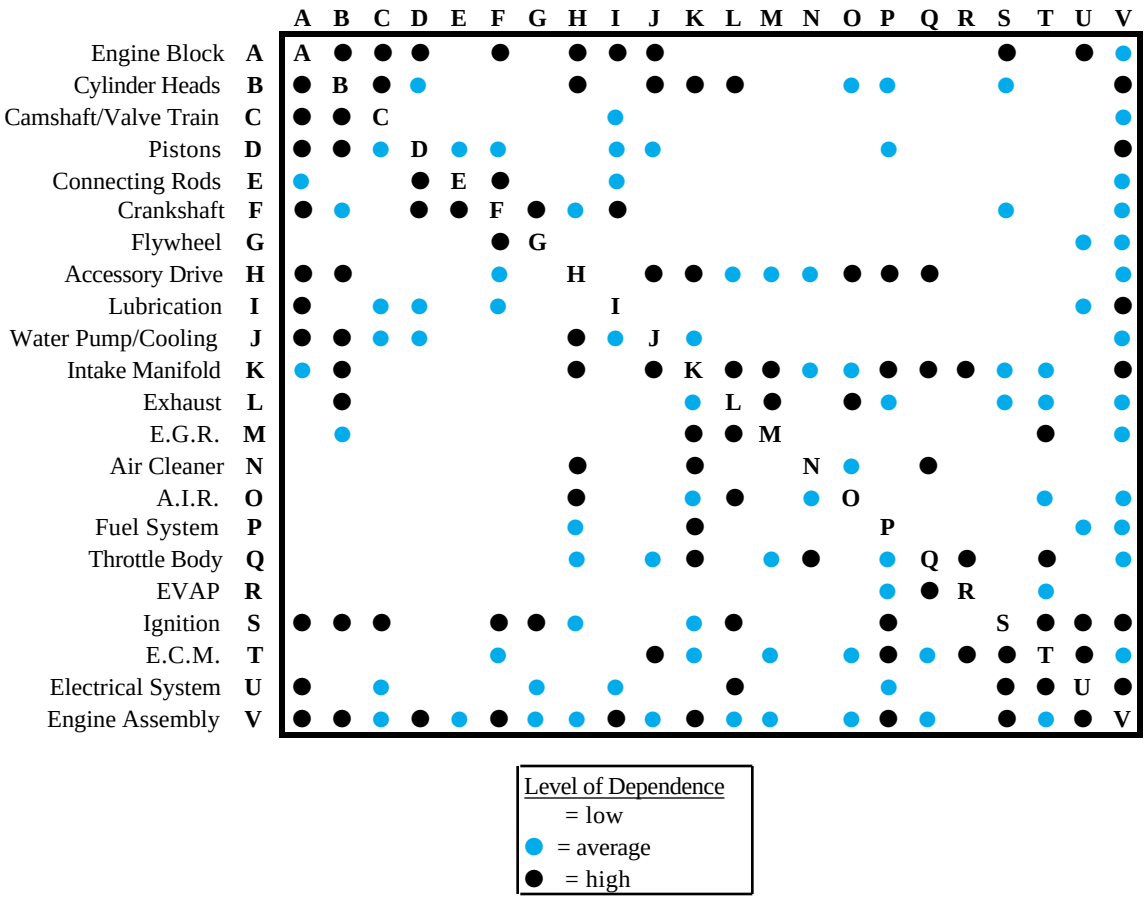

Figure 3: Example DSM—Information Flow Between PDTs

Rearranging the rows and columns of the DSM as in Figure 4, we now clearly see a sensible organization design with four system teams that potentially facilitates interteam integration. Some of the PDTs are members of more than one system team. Two of the PDTs, B and K, are broken into two teams each to remove (decouple) some of the interdependencies. Meanwhile, the five PDTs at the bottom of the matrix (H, S, T, U, and V) did not fit neatly into the four system teams. These PDTs need to interface heavily with all the system teams. Hence, these PDTs are integrated at the program level with different IMs. (Perhaps a representative from each of these PDTs will be assigned to attend each of the system teams' meetings.)

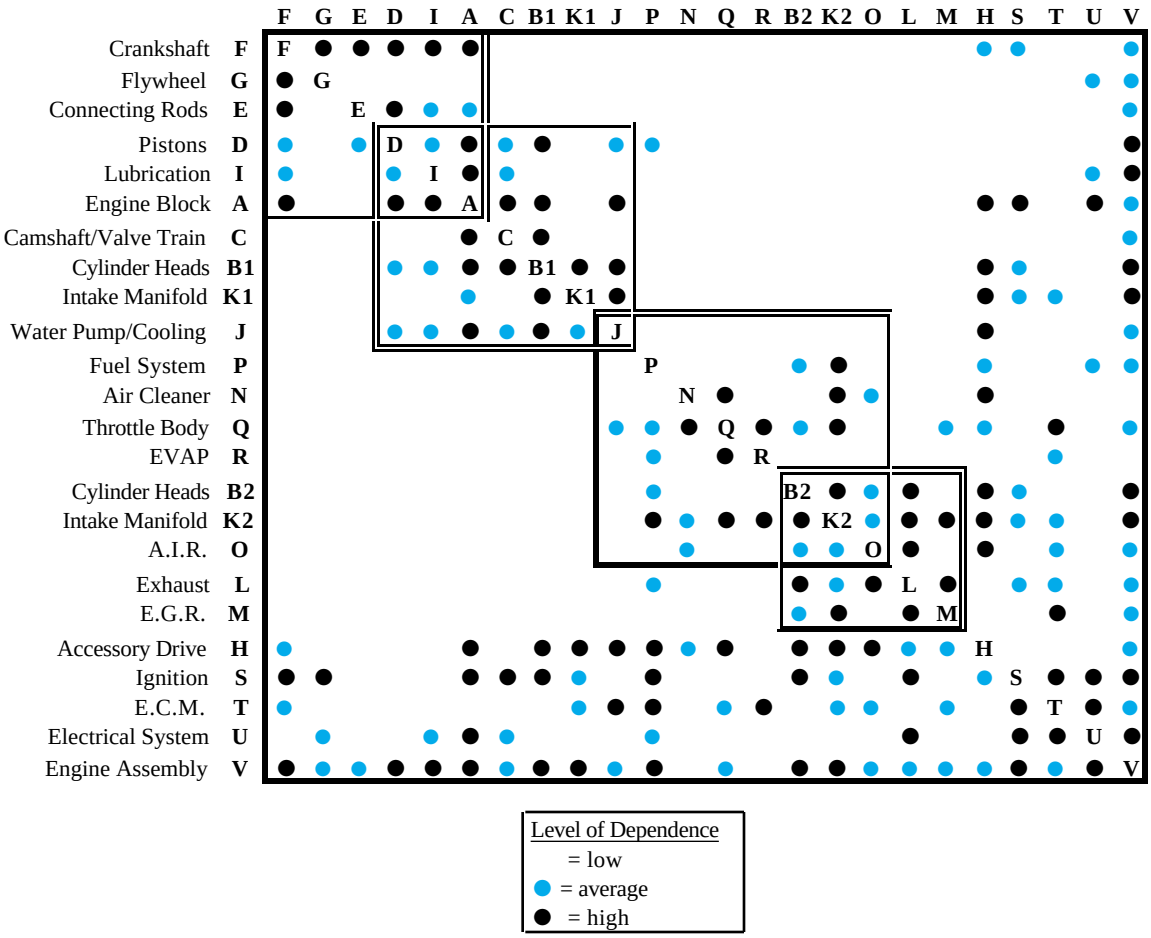

Figure 4: Rearranged DSM Clarifies Organization Design

This research applies the people-based DSM to a section of the F/A–18E/F program. While doing justice to an analysis of this nature would be a project unto itself, the first-look examination reinforces the utility of the DSM as an organizational analysis tool for the defense aircraft industry. With a completed DSM before them, individuals can visualize an entire program in a succinct format and ask questions from a system perspective—all the while working from a common tool. Moreover, having everyone view an issue in a new format can help level the input-generating playing field, stimulating new ideas and approaches. People-based DSMs can foster creativity in addressing the following questions:

- Do teams differ in their perceived levels of interaction? Why? Perhaps they do not fully recognize their information suppliers and customers. One team may see itself merely as an information provider or recipient, unaware of exactly how its inputs figure into the overall organization.
- A DSM provides insights into where information flow bottlenecks or other difficulties may occur. It can help keep information from having to be pushed or pulled.

- Teams responsible for providing high levels of information could be front-loaded in their budgeting profiles, if necessary, to help ensure that this information is available when needed. A manager can ask if the current budget profiles make sense based on the data in the DSM. Would providing additional resources to the information sources reduce process risk?
- Should the teams be organized differently? How should teams with high bandwidth interfaces be integrated? Where might particular IMs be especially necessary? Certain interfaces might indicate the need for co-location, special software tools, common databases, etc. Certain interactions might proceed more efficiently with the aid of a liaison or zone engineer of some sort, or when overseen by an integration team.
- Are already present IMs applied judiciously? Are the liaisons, zone engineers, integration teams, etc. as they now stand appropriately distributed? Are the current plans for software tool, common database, co-location, and other improvements well founded from an overall organization perspective?
- Managers should know, based on their own experiences with the teams, whether or not the DSM picture makes sense. If a high level of information flow is shown, has this really been observed? Are the interactions noted the ones that *should* be taking place, based on knowledge of the system as a whole? Are any interactions surprising—ones that may not have been considered before?

6. Key Findings from Five Case Studies of IMs

Examination of the use of IMs in five cases supports analyses found elsewhere in the literature. The result boils down to a collection of principles and recommendations regarding the appropriate uses of particular IMs. They represent the accumulated best practices and lessons learned from the five programs studied and numerous others documented in the literature. Nevertheless, the difficulty in reaching universal conclusions should be evident. Different tasks and circumstances can require varied tools. Integrators should be aware of the strengths and weaknesses of each IM, complementing this with a broad knowledge of the system product and the organization's culture and traditions.

1) Systems Engineering and Interface Optimization

- As much as possible, design the organization to mirror the product architecture. A well partitioned system will have a minimal number of intersubsystem interfaces. Therefore, the teams developing these subsystems will require a minimal amount of interaction.
- Maximize the ability of teams to communicate while minimizing their need to do so.
- Meanwhile, do not let teams get "tunnel vision" and disregard their external interfaces.
- A systematic approach works best for grouping the IPTs and FSGs into system teams and for determining how integration will occur within and between the levels. People-based DSMs provide a useful tool in this regard. System models used for these purposes make useful IMs themselves.
- None of the programs investigated utilized a systems approach to designing and integrating the organization, although some programs had groups in place to enable learning so future programs would have this opportunity.

2) Improved Information and Communication Technologies

- E-mail can lead to overcommunication: people copy messages to large distribution lists. Most people do not have the time to assimilate all the messages. It would behoove programs to establish some formal guidelines for the use of e-mail if they are going to rely on it as an IM.
- Common databases should span high bandwidth interfaces (in the system architecture and in the organization). Common databases require accessibility, decreased access time, standardized data formats, and user training. All data relevant to design objectives, including cost data, should be readily available. Intranets and browser software provide high utility access.
- Critical system, subsystem, and element parameters such as cost and weight should be tracked regularly versus design goals and made obvious to everyone.
- Standardization of software facilitates integration but can become a tradeoff with innovation and flexibility.
- Seamless electronic file transfer is essential. LANs and intranets are good, fast options. E-mail is sometimes an option as well. Security issues are barriers; encryption needs to be available and simple to use.
- CAD/CAM/CAE systems are critical, providing a common reference point for cross-functional designers. Yet, formal guidelines are often necessary to achieve optimal integration.
- Integrated scheduling leads to improved multi-team integration.
- Other successful communication IMs include: standardized data sheets and electronic worksheets, data language standards, archives of lessons learned, well-organized process guides (in conjunction with training and incentives), and bulletin boards.
- Note that expedient communication IMs—e.g., teleconferencing—tend not to enable easily accessible and searchable documentation.

3) Training

- Develop and administer training up front in the program, even as a carefully contained critical path item.
- Integrative training provides interteam role, responsibility, and contribution awareness.
- IPT-building training is best experienced by the entire, multi-disciplined team together.
- Equip the teams with a common understanding of the program's interteam interfaces, in terms of data flow, goals, and priorities.
- Establish and raise awareness of documentation protocol.
- Like information, training should be available to the right people, at the right place, and at the right time. Missing these marks quickly diminishes the value added.

4) Co-location

- Co-location is an excellent IM, although many do not utilize it in its most effective form. Usefulness is greatly diminished when group members are more than 10 meters apart. Hence, for large programs, correctly choosing which groups to co-locate is crucial. Systematic methods, such as the people-based DSM, facilitate making appropriate co-location choices.
- Specialized test labs, manufacturing facilities, multi-company teams, etc. constrain co-location.
- Greater co-location can reduce the need for other IMs, while a lack of co-location requires special consideration of other IMs.

5) "Town Meetings"

- Most intuitively agree that town meetings have a positive effect if they are well-planned and emphasize integrative concepts.
- How often? Once a month, once a quarter, and after significant milestones are popular options.

6) Manager Mediation

- Hierarchical management structures are *not* the most efficient way to arbitrate and mediate all interteam issues.
- Management must provide a clear vision of program direction, ensuring that knowledge of that direction is held broadly within the program.

  Strategy => Goals => Incentives => Good low level decisions

- The role of a *heavyweight project manager* (HPM) is difficult to implement on a program-wide basis in large, complex programs, but may apply at the system team level.
- The most practicable management-related, interface mediation mechanism seems to be management teams (similar to integration teams—see IM #***8***) composed of managers from multiple functions and/or teams.

7) Participant Mediation

- Liaison roles have positive effects: the rapid communication of information, the opportunity to experience and learn from the workings of other teams, and the removal of some of the interface management burden from the team leader.[10]
- Zone engineers oversee interteam issues in the F/A–18E/F program.
- IPTs can designate members to handle specific interfaces.
- A good interteam representative possesses a thorough understanding of and respect for the tasks and goals of both or all of the teams with which he or she works.
- A participative mediator needs a systems view of the issues and a clear conception of the overall strategy that provides guidance for the trades that must be made.
- Guidelines should establish the amount of information filtering the liaison will provide.
- Liaison-type roles fall along a spectrum, with authoritative roles at one end and simple information transmitters at the other. The amount of power and responsibility bestowed on a liaison will vary depending on the character of the interface they oversee.
- Overlapping team membership is another (informal) form of participant mediation, but employees should not be part of more than two teams at once.

[10] (Sheard 1995), p. 2

8) Interface "Management" Groups and Integration Teams

- Most of the five programs studied use a team approach to interface oversight and mediation.
- These teams consist of both managers and participants.
- While integration teams exist at a higher level than IPTs in a systems hierarchy sense, they need not necessarily consist of individuals with a higher rank in the organization.
- Integration teams can be proactive and anticipatory to varying degrees. The most successful ones are very clear on their roles, resources, and responsibilities.

9) Interface Contracts and Scorecards

- The documentation effort required to establish and maintain interface contracts can be extraordinary. They are most crucial at the outset of a program, to raise awareness about key interactions. After that, maintaining them should not be allowed to consume as much effort.
- Scorecards provide a valuable way of evaluating interteam interactions. A scorecard is only as good as the metrics it notes and the ability of the user to evaluate them.
- Without a clear understanding of the information that needs to flow across an interface, its appropriate frequency, and the ease of transmission, proper scoring will be difficult.
- The process of developing helpful scorecards will in itself enlighten their users to the characteristics of effective interteam interfaces.

The above findings regarding the appropriate uses of IMs help guide organization designers and program management in the enlightened choice of integration options.

7. The Design for Integration (DFI) Process

A systematic approach to program integration can go a long way towards creating effective interfaces between teams and ensuring the proper flow of information. Mohrman *et al.*, having looked at team-based organizations extensively, note that "the most effective teams we saw used systematic planning processes for determining responsibilities and for scheduling and integrating their work."[11] Planning and forethought regarding IPT interfaces likewise facilitates effective program execution. This section outlines the key steps in a proposed approach to *design for integration* (DFI). Note that "design for x" in this sense applies to the design of the design process. While additional details, unique to a given program, will provide constraints and guidance in this process, the general steps shown in Figure 5 apply to the majority of situations.

[11] Mohrman *et al.*, p. 178

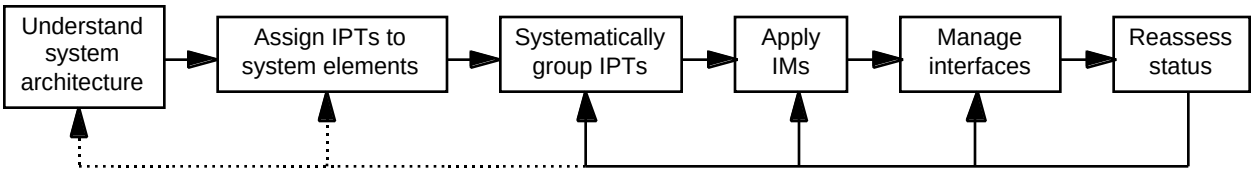

Figure 5: The DFI Process

As the control loop implies, the DFI process is iterative. Each of these steps is elaborated below, along with suggestions for implementation.

1) Know system architecture

The first step for a program's organization designers is to possess a thorough understanding of the architecture of the system to be produced. The effectiveness of the decomposition of the complex system's required functions into well-partitioned subsystems and elements drives the ease of work allocation and team integration.

Given an unprecedented system, however, possessing a thorough understanding of the architecture is understandably difficult. The need to better understand system specifications has thus brought the recent focus on requirements management and on systems engineering as a discipline. Knowledge of tasks and their duration is essential to the creation of the *statement of work* (SOW),*work breakdown structure* (WBS), and program scheduling. It is likewise crucial for interface planning and management. System models can greatly facilitate this effort. Where the architecture and/or tasks are yet to be determined, organization designers must build in flexibility so the organization can adjust once the characteristics of the required IPT and system team interfaces settle out. Baseline organization designs for programs developing unprecedented systems will have to be the most flexible of all. (This requires an incentive system that does not promote resistance to organizational flexibility.)

2) Assign IPTs to system elements

Once the architectural subsystems and elements have been defined to the extent possible, an IPT should be assigned to the development of each element. Ideally, each IPT will include all of the cross-functional and other resources necessary over the life of its task, although constraints limit this possibility and will create the need for additional external interfaces. Different levels of task complexity between interdependent teams can lead to interface difficulties. For example, one team might have a "big picture" perspective while its sibling IPT dwells in the details. (This research does not detail the creation and management of IPTs themselves, although this is a salient

issue and the subject of much other research.) In some cases, it may be more appropriate to assign some tasks to FSGs.

3) Systematically group IPTs

Once IPTs have been formed to develop elements of the system architecture, the IPTs and necessary FSGs should be grouped into system teams. System teams will tend to form around major subsystems, especially if the architecture is wisely decomposed and partitioned. However, constraints affecting this decomposition are likely to differ from those in the organization. It is best to conduct an additional analysis with a people-based DSM to make the system team grouping determination from the "bottom up."

Thus, system teams must be "built" as carefully as IPTs. They need charters, just like IPTs. It may be necessary to provide team-building training to each system team's constituent members. In addition, appropriate management and leadership roles for the system team and program must be determined.

Grouping the IPTs and FSGs into system teams is a key step in the DFI process, for it determines which interfaces will be crucial for the IPTs, FSGs, and system teams in the program. Desirable information transfer interfaces should be:

- **Defined**, in terms of what information needs to flow, where, when, and how (i.e., "the right information at the right place at the right time").
- **Tight-fitting**, in terms of task assignment. Tasks should not overlap or "underlap."
- **Permeable**, in terms of permitting and regulating information flow. Information should arrive "just-in-time"—not too early or too late. It must be the correct amount of information—not more and not less. It must flow readily and smoothly, yet not inundate its recipients. The interface must allow just the right amount of the right information to flow.
- **Mutable**, in terms of altering information flow. There must be a means of adjusting what information gets transferred, when, and how.
- **Efficient**, in terms of time lag from provider to recipient. This path should be free of undue bureaucracy or other delays.
- **Documented**, in terms of keeping a record of information flow. Information useful once may be useful again. To avoid "reinventing the wheel," one needs a record of the flow. Documentation facilitates learning and accumulation of a knowledge base.
- **Measurable**, in terms of allowing analysis of success and flow rate. Success should be based on objective criteria where possible. Metrics to evaluate information flow and interteam interactions are crucial to provide appropriate incentives and further process improvement.
- **Adapted**, in terms of the program's task, size, and stage. One size does not fit all. Each important interface deserves explicit, personalized, unique attention and optimization.

The goal of this DFI step is to come up with the best system team breakouts and the beginnings of a viable *scheme for interface mediation* (SIM) for the program. A SIM document explicitly outlines the expected interfaces, their desired characteristics, and the means of monitoring them. This is also the stage where resources that are not available to each individual team are included at the system team or program level. A program's success will be affected by how well the system team and integration level determinations are made.

4) Apply IMs

Once IPTs, FSGs, and system teams have been determined and interface characteristics have been established, the appropriate IMs must be implemented to facilitate the interactions—based on the unique physical, political, architectural, and other attributes of the program. Findings such as those in §6 will be helpful here. The deliberations in this step—especially the reasons for choosing particular IMs—should be recorded in the SIM document so they can be communicated to the entire program and revisited later (Step ***6***).

5) Manage interfaces

After designing the best possible organization structure and choosing appropriate IMs *a priori*, program management must keep information within the program flowing smoothly. This includes mediating technical issues via suitable IMs and monitoring the effectiveness of communication. Evaluating the effectiveness of the SIM is difficult on a real-time basis. Often only "lagging indicators" such as adherence to budget and schedule on a macro level, are available as proxies. Integrated subsystem prototype testing and design reviews also provide opportunities to formally evaluate organizational integration effectiveness. Are the issues that result from these tests and reviews the result of a lack of information? miscommunication? unresolved technical issues? Checklists and scorecards can assist reviewers in asking the right questions about integration. Tracking the time spent (perhaps through charge numbers) by members of integration teams and other individuals on IM-related tasks might also yield interesting data on which mechanisms are being utilized and where the issues reside.

6) Reassess status

As the program evolves, the SIM will need to be reevaluated. Some IMs will no longer be appropriate. Some interfaces will become more important, others less. New interfaces will form. Some will disappear altogether. As the organization changes, so must the IMs that connect it. It should be the periodic, if not ongoing, task of a group of organization designers to reevaluate the SIM and effect the necessary changes. Making sure this process is well-documented will greatly

improve its effectiveness in the long run, especially for long-term development projects where turnover among the membership of the SIM group is likely.

8. Lean Principles of Interteam Integration

This research illuminates several principles or enabling practices pertaining to interteam integration. These principles should be included as part of the evolving conception of effective, lean IPD:

- **Proactively design programs for integration (DFI process)**
  Efforts should be made up front in a program to design an organization of IPTs, FSGs, and system teams based on a systematic approach, such as one bearing in mind interteam information flow requirements. Organization structure should not be determined without explicit consideration of the interfaces between teams and how they will be managed (i.e., which IMs will be used). Enlightened decomposition of the system product architecture, coupled with an organization that mirrors that architecture, will also circumvent many potential integration issues.
- **Recognize the utility of information flow modeling in DFI (people-based DSM)**
  Information flow modeling provides a systematic basis for organization design. It allows for explicit consideration of interteam interactions and can help illuminate interfaces requiring specialized IMs.
- **Explicitly consider the appropriate uses of IMs**
  All IMs are not equally appropriate at a given time or in a given place. An understanding of desired interface characteristics will help one find the right IM for the job.
- **Systematically evaluate the multi-level integration tradeoff**
  The level in the organization at which particular functional representatives, resources, and perspectives are integrated should be informed by a systematic analysis of interface requirements and knowledge of relevant constraints. (While constraints must be accommodated, the most limiting ones might become the target of special initiatives.)

Related to these principles and enabling practices are others identified in the *Lean Enterprise Model* (LEM) under development by LAI. The table below summarizes how IMs, people-based DSMs, and the DFI process (which includes the former two columns) enable each of the overarching practices noted by the LEM. Numbers in the table refer to explanatory notes that follow.

LEM Overarching Practice	**IMs**	**People-based DSM**	**DFI process**
Identify and Optimize Enterprise Flow	1	2	3
Assure Seamless Information Flow	4	5	6
Optimize Capability and Utilization of People			7
Make Decisions at the Lowest Possible Level			8
Implement Integrated Product and Process Development	9	10	11
Develop Relationships Based on Mutual Trust and Commitment	12		
Continuously Focus on the Customer			13
Promote Lean Leadership at All Levels			14
Maintain Challenges of Existing Processes			15
Nurture a Learning Environment	16		
Ensure Process Capability and Maturation			17
Maximize Stability in Changing Environment			

Table Notes

1. Enterprise flow applies just as much to *information* in the design process as it does to product parts in the factory. Many have characterized the design process as one of information flow. IMs facilitate the flow of information through organizational channels.
2. People-based DSMs identify information flow patterns and can be analyzed to optimize that flow.
3. The DFI process seeks to identify and optimize program information flow by using systematic methods (DSMs) and appropriate IMs.
4. IMs streamline information flow through the multiteam organization.
5. People-based DSMs act as an IM for interface minimization, allowing teams to concentrate on the important information transmissions.
6. The DFI process designs organizations on the basis of information flow optimization.
7. The DFI process forces organization designers to explicitly consider the level of integration for various functional, customer, supplier, and other representatives so as to maximize their effectiveness given relevant constraints.
8. By stressing integration at the lowest level possible given constraints, the DFI process encourages decision making at the IPT (system element development) level while ensuring as global a design decision as possible by highlighting the key interfaces for that group.
9. IMs are the “glue” of IPD.
10. People-based DSMs highlight where the “glue” (in 9.) must be applied, and in what amounts.
11. Designing a program for integration is the surest way to enable IPD.
12. Several of the IMs studied serve to boost camaraderie and interteam awareness and respect. Mutual respect and commitment is of course important for multiple teams just as it is for customers and suppliers.

13. The customer representatives are included in the integration needs analysis. The DFI process incorporates the voice of the customer (i.e., actual representatives) at the appropriate organizational level.
14. Lean leadership is enabled in part by quality knowledge of the suppliers and customers of information important to the task at hand. The DFI process facilitates the clearer definition of leadership roles and responsibilities.
15. The DFI process is an iterative one requiring continuous improvement.
16. IMs are considered with a view towards documenting and disseminating important decisions and lessons learned. Common databases, archives, process guides, and information gatekeeper roles serve to build knowledge bases. IMs also seek to establish cross-functional awareness and encourage workers to move towards a systems view.
17. Mastering the DFI process constitutes an important step towards establishing a consistent IPD process.

Together, the principles and enabling practices recognized above enhance the IPD paradigm by assimilating concepts of improved information flow via interteam integration and the use of appropriate IMs.

9. For More Information

Browning, Tyson R. (1996a) *Systematic IPT Integration in Lean Development Programs* Unpublished Masters Thesis, M.I.T.

Browning, Tyson R. (1996b) "A Systems Engineering Approach to Multi-Team Integration: Interdependence and Integrative Mechanisms" *Proceedings of the Sixth Annual International Symposium of INCOSE*(July): 8 pages.

Browning, Tyson Rodgers (1997) "An Introduction to the Use of Design Structure Matrices for Systems Engineering, Project Management, and Organization Planning" M.I.T. Lean Aircraft Initiative, Working Paper

Cole, Willie E. (1995) "Cooking Up a Batch of Team Synergy: Ingredients for Setting Up Successful Teams" *Program Manager*(Sept.-Oct.): 28-33.

Eppinger, Steven D., Daniel E. Whitney, Robert P. Smith, and David A. Gebala (1994) "A Model-Based Method for Organizing Tasks in Product Development" *Research in Engineering Design* **6**: 1-13.

McCord, Kent R. and Steven D. Eppinger (1993) "Managing the Integration Problem in Concurrent Engineering" M.I.T. Sloan School of Management, Working Paper no.3594.

Rechtin, Eberhardt (1991) *Systems Architecting: Creating & Building Complex Systems*. Englewood Cliffs, NJ, P T R Prentice Hall.

Sheard, Sarah A. and M. Elliot Margolis (1995) "Team Structures for Systems Engineering in an IPT Environment" *Proceedings of the Fifth Annual International Symposium of INCOSE*.