Regular Paper

Designing System Development Projects for Organizational Integration

Tyson R. Browning

Lockheed Martin Corporation, P.O. Box 748, MZ 2222, *Fort Worth*, TX 76101 (*e-mail*: *tyson@alum.mit.edu*)

Received June 1, 1999; revised August 24, 1999

ABSTRACT

In complex system development projects, *integrated product teams* (IPTs) may be used to develop various system components. Many have struggled to determine the characteristics of highly effective IPTs and the circumstances in which particular perspectives should be incorporated within an IPT. However, much less research has addressed the nature and management of the relationships between IPTs—the integration of IPTs within a project. While many have lamented that coordination problems have played a large part in diminishing the performance of their projects, a systematic approach for considering these issues *a priori*, during the organization design activity, is lacking. This paper describes a framework for thinking about organizational integration within a project and develops a systematic approach for designing organizations, *design for integration* (DFI), that explicitly considers integration needs. © 1999 John Wiley & Sons, Inc. Syst Eng 2: 217–225, 1999

1. INTRODUCTION

Developing complex systems requires a myriad of perspectives and a diversity of expertise. All of these contributions must be organized and coordinated. Organization design entails prescribing an organization structure that enables specialized people and groups to coordinate their work effectively. In the last decade, concurrent engineering and *integrated product development* (IPD), implemented through the use of *integrated product teams* (IPTs), has guided organization design in complex system development projects. IPTs bring cross-functional (discipline) and upstream/downstream (process) representatives to bear on the development of particular system components. Many have struggled to determine the characteristics of highly effective IPTs and the circumstances in which IPTs should incorporate particular discipline and process representatives—i.e., relationships *within* IPTs. However, much less research has addressed the relationships *between* IPTs. Yet, many organizational integration problems result from poor coordination between IPTs, and these problems contribute significantly to diminished project performance. A systematic approach for considering these complications up front, as part of organization design, is lacking.

The realm of organization design benefits from systems engineering concepts. Indeed, integration challenges in the organization stem from systems architecting and engineering issues. A system derives value from the relationships among its parts. These interactions make a system much greater than its parts. Rechtin [1991: 29] finds the greatest leverage in system architecting at the interfaces. While true for a product system, this tenet also applies to organization "systems." Like a well-partitioned and well-integrated product architecture, a well-partitioned and well-integrated development organization can provide a competitive advantage. Moreover, these two systems are related: The partitioning of the product architecture affects the partitioning and integration of the development organization. Organizational interfaces reflect architectural interfaces.[1] Recognizing the connection enables the discerning organization designer to exploit architectural decisions to enhance interteam integration.

Complex system development implies complex organizations. Complexity manifests itself as numerous, highly coupled components. Thus, effective organization design and integration becomes harder as system complexity increases. IPTs working together to develop complex systems face a daunting task. They depend on each other for information (sometimes without realizing it). They must interact. Tausworthe cautions, "A team producing at the fastest rate humanly possible spends half its time coordinating and interfacing."[2] Without cautious system partitioning, the number of required interfaces increases drastically with system complexity. IPTs with multiple interdependencies get bogged down with coordination efforts and have to redesign part of the system when their original work is based on poor information. Cole [1995, p. 30] notes, "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." IPTs must interact because the work required of each IPT depends upon the work of at least one other group.

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" [Eppinger et al., 1994: 1]. Understanding and handling interface issues would seem to hold great potential for improving organizational design and integration. Organizational integration is a project management issue, but it can benefit from a systematic approach based on systems engineering principles and the architecture of the system being developed.

In the efforts to create effective interfaces between teams and ensure the proper flow of information, a systematic approach helps guarantee the inclusion of important considerations. While organization design and integration are often handled in an intuitive fashion, a systematic approach has several advantages. Systematic approaches introduce rigor and structure to the decision-making process. They serve as a baseline (not as a substitute) for intuitive judgments. They lend themselves to comparison and benchmarking, making change measurable. Furthermore, systematic methods provide an orientation framework that the varied perspectives addressing the issue can agree on and work from. Mohrman et al. [1995: 178], 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." Systematic planning and forethought regarding IPT interfaces can facilitate effective project execution. This realization has spawned an emerging field called enterprise engineering.

Effectively managing organizational interfaces requires (1) ensuring the proper interactions between well-partitioned organizational components and (2) facilitating the smooth transfer of information across interfaces. The right teams must be formed, the right interfaces must be arranged between them, and the right *integrative mechanisms* (IMs) must streamline the necessary transfers across those interfaces. Along these lines, this paper describes a framework for thinking about organization integration within a project. The paper then outlines a systematic approach for organization design, *design for integration* (DFI), that explicitly considers integration needs.

2. A FRAMEWORK FOR ORGANIZATIONAL INTEGRATION

Complex system development projects typically utilize a hierarchical organization structure. The hierarchy abstracts the web of interactions between organizational components based on a partitioning. In such a structure, integration must occur at several levels, both within IPTs and between IPTs. Hence, at least two levels of integration exist, internal and external to an IPT. Furthermore, it is not practical or feasible that all activities will be accomplished using IPTs. Various *functional support groups* (FSGs) such as test labs may remain outside the IPTs. Yet, certain IMs can successfully integrate FSGs at a higher level.

[1]Hence, some authors [e.g., Grady, 1994] recommend the organization structure mirror the system architecture.

[2]Quoted in [Rechtin, 1991: 284].

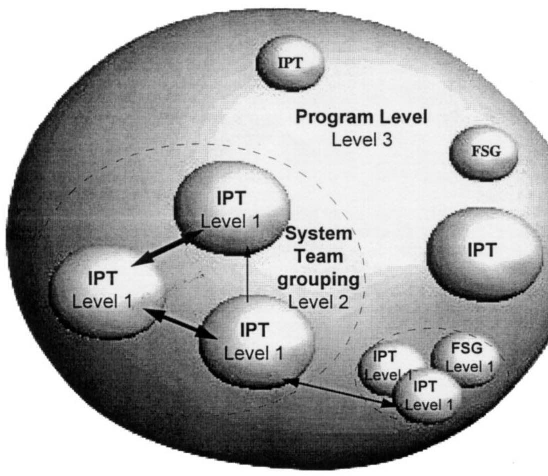


Figure 1 Three levels of integration.

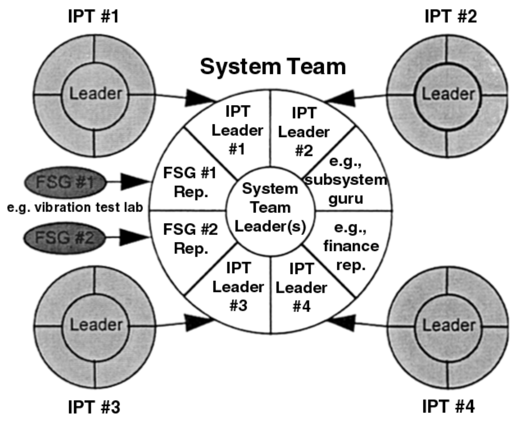


Figure 2 Example system team composition.[4]

In light of this possibility, it is convenient to think of organizational integration as necessary at three (or more) levels. McCord and Eppinger [1993] use the term "system team"[3] to indicate a set of IPTs and FSGs aggregated based on mutual, high traffic interfaces. Thus, IPTs represent a first level of integration; system teams exemplify a second; the overall project becomes a third. Figure 1 illustrates the three-level framework. Individual IPTs (and FSGs, if applicable) can be grouped in system teams, which, along with other IPTs (and FSGs) and system teams, comprise a project or program. IPTs and FSGs that do not fit within a system team are integrated at a higher level. In Figure 1, the arrows represent the most significant information exchange needs; the dashed ovals distinguish system team groupings.

The three level structure in Figure 1 provides a minimum amount of hierarchical distinction. Large projects may require additional levels between one and three—e.g., different levels of system teams. For example, a large, military fighter plane project (the F/A-18E/F program) at one time used five levels to capture finer distinctions between system team sizes and/or priorities.

Figure 2 provides an example of system team composition. This system team integrates four IPTs and two FSGs as well as a couple of functional perspectives not represented at the IPT level. The team includes FSG and IPT representatives (here the IPT leaders), additional functional representatives, and one or more system team leaders (who represent the system team at the project or higher system team level).

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 a higher level? Addressing these tradeoffs and appropriately applying IMs to facilitate inter-IPT and system team integration is the subject of the proposed DFI process.

3. TOWARDS A SYSTEMATIC APPROACH TO DESIGNING PROJECTS FOR ORGANIZATIONAL INTEGRATION

This section outlines the key steps in a proposed approach to *design for integration* (DFI). DFI applies to organization design, and it explicitly considers interteam integration. The six general steps shown in Figure 3 apply to the majority of situations. Details unique to a given project will provide additional constraints and guidance. The six steps do not constitute *the* DFI process, but *a* DFI process. This example illustrates important considerations in DFI efforts. The process emphasizes the necessity of explicitly and systematically contemplating integration issues when designing organizations.

As the control loop in Figure 3 implies, a DFI process is iterative. The dashed portion of the feedback loop indicates changes that will most likely take effect on a subsequent project, whereas the solid portion represents easier opportunities to modify ongoing projects. Each of these steps is elaborated below, along with suggestions for implementation.

[3]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 components of a particular subsystem—one of many in the overall system which the project as a whole is developing.

[4]This diagram is adapted from PRTM's illustration of what they call Interlocking Development Teams. While the illustration is useful here, the related methodology is different.

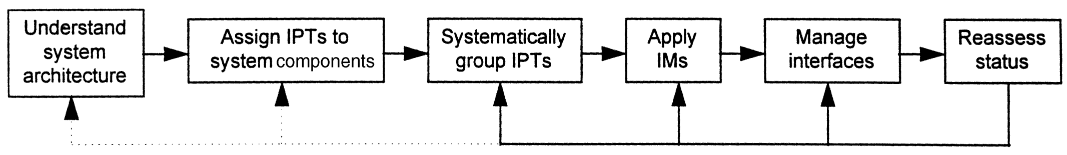

Understand system architecture
Assign IPTs to system components
Systematically group IPTs
Apply IMs
Manage interfaces
Reassess status

Figure 3 A design for integration process.

3.1. Understand System Architecture

Gulati and Eppinger [1996] explore the coupling of product architecture to organizational structure, noting that decisions in one realm affect and even constrain opportunities in the other. Figure 4 shows the proposed relationship: Architecture and organization are linked through the generalized inverse exercises of system decomposition and integration. When considering organizational integration issues, one must look at the architecture of the system that the organization will develop. The first step in the DFI process is to understand as completely as possible (or practical) the nature of the system architecture, especially its decomposition and the resulting interfaces, for these will directly affect the organization and the ease of integrating the teams working on the various subproblems.

How, specifically, does a product architecture affect the organization and its communication patterns? Morelli, Eppinger, and Gulati [1995] investigate the extent to which coordination-type communication between project groups is predictable given a known task set based on a system architecture. With the proposed tasks before them, project participants were able to predict 81% of the communication that in fact took place. This result signifies the possibility of designing an organization based on a proposed system architecture and its associated task set.

Given an unprecedented or revolutionary system, however, possessing a thorough understanding of the architecture and the tasks necessary to develop it is difficult. The need to organize and develop that understanding has brought attention to the discipline of systems engineering. Knowledge of tasks and their duration is essential to the creation of the *statement of work* (SOW), *work breakdown structure* (WBS), and

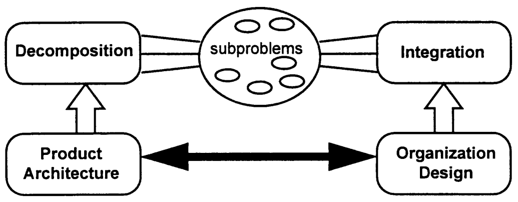


Figure 4 Architecture tied to organization through decomposition/integration problem [Gulati and Eppinger, 1996].

integrated master schedule (IMS). System architectural knowledge is likewise crucial for interface planning and management. For upgrades and other largely precedented systems, it is much easier (although not necessarily easy) to outline tasks and their information requirements. Where the architecture and/or tasks remain to be determined, organization designers must build in flexibility so that the organization can adjust once the characteristics of the required IPT and system team interfaces settle out. Baseline organization designs for projects developing unprecedented systems will have to be the most flexible of all. (This requires an incentive system that encourages organization flexibility.)

The issues of systems architecting are crucial as inputs to any design process. The importance of intelligently grouping functional requirements and allocating architectural elements to meet those requirements cannot be overstated. It is here that DFI begins, for it is here that much of the difficulty of the remaining integration tasks is determined. However, this paper will not address the tenets of intelligent systems architecting and decomposition, which several authors have discussed [e.g., Alexander, 1964; Altus, Kroo, and Gage, 1995; Baldwin and Clark, 1999; Kusiak and Wang, 1993; Michelena and Papalambros, 1995; Pimmler and Eppinger, 1994; Rechtin, 1991; Reinertsen, 1997; Sanchez and Mahoney, 1997; Smith and Reinertsen, 1991]. Leaving aside the notion that product architectures can be designed with a consideration of organization integration concerns, the remainder of this paper will focus on how, given an architecture, one might proceed to design an organization to develop it.

3.2. Assign IPTs to System Components

Once architectural subsystems and components have been decomposed and defined to the extent possible, an IPT is assigned to the development of each component. Ideally, each IPT would include all of the cross-functional and other resources necessary over the life of its component development activity. However, resource constraints limit this possibility and will create the need for additional external interfaces. In addition, depending on the size of the project, and realizing that IPTs should be kept to a small size, one must determine what scale of component the IPT will develop. These issues and others combine to make the IPT assignment problem nontrivial.

An important consideration, IPT size, directly constrains integration efforts. Since IPTs include cross-functional (discipline), upstream/downstream (process), and perhaps even customer and supplier representation, they require more people than an equivalent, functional team. Klein and Susman [1995] found the average size of an IPT in the defense aircraft industry to be 40 members (26 full-time and 14 part-time). However, Katzenbach and Smith [1993: 45] set the size of an ideal team at close to ten people, within a range from 2 to 25 people. Peters [1995] notes how the reorganized Space Station project uses IPTs of 8–12 members. Sheard and Margolis [1995] cite D. Quinn Mills as putting the "ideal" team size at 5–7 people. These ranges represent the number of people who can typically work together as an effective team "should." Larger groups can still be called teams, of course, but they will find it difficult to achieve the same level of intimate teamwork espoused by the proponents of "teaming." As Cole [1995, p. 30] understands, "Group dynamics being what they are, groups of say, 30 or more, tend to break themselves down into smaller groups anyway." Certainly, teams with 70 or 100 members will not be able to perform as IPTs should, and should not be termed as such. Therefore, given the size limits for effective IPTs, one is automatically constrained to assign them to a task of approximately subassembly level development scope.

In its first column, Table I exhibits the architectural hierarchy nomenclature used in space systems development [Rechtin, 1991; Shishko et al., 1995]. The second column shows an example mapping to organizational structures. Large projects may utilize all of the listed levels; smaller projects may not. Quite a bit of flexibility exists as to the level of correspondence between architectural and organizational components. Larger projects will likely require multiple levels of system teams. In this case, the distinction between system teams and IPTs may blur. Since many resources and perspectives can only be integrated at higher levels, some refer to system teams as "high level IPTs." While the distinction does not have to be made, one should keep in mind that the intent of the IPT is to integrate diverse perspectives at the lowest level possible. The step of assigning IPTs to appropriately scoped tasks precedes the step of integrating those IPTs in system teams. As DFI Step 3 describes, system team structures should stem from known IPT task assignments and their more predictable interfaces.

Table I. Mapping Architectural to Organizational Hierarchies (example)

Architecture	Organization
System	Project
Segment	
Element	
Subsystem	System Team
Assembly	
Subassembly	IPT
Part	

Examining the activities required to develop particular components can provide additional insight when assigning tasks to IPTs. Activity interdependencies affect processes and workflows and impact IPT interdependencies, composition, and appropriate level of integration. 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. Finally, it may be appropriate in some cases to assign certain tasks to functional teams or FSGs.

Much more could be said about chartering teams. This paper does not focus on the creation and management of IPTs, although this is a salient issue and the subject of much other research. Instead, the point here is to show how this important step fits into a DFI process.

3.3. Systematically Group IPTs

After IPTs and FSGs have been assigned to develop low level components of the system architecture, these groups should be integrated into system teams. System teams will tend to form around major subsystems, especially if the architecture is wisely partitioned. However, constraints affecting architectural decomposition may differ in some ways from those impacting the organization. Hence, it is best to conduct an additional analysis to compose the system team from the "bottom-up" after the formation of lower level organizational components. This exercise has been termed "integration analysis" [Eppinger, 1997] and "strategic grouping" [Nadler and Tushman, 1988]. McCord and Eppinger [1993] and Fernandez [1998] apply a *design structure matrix* (DSM) and clustering algorithms to this problem.

Another important consideration at this stage in a DFI process is at what level to integrate certain teams and resources. Resources unavailable to individual teams are included at the system team or project level. When a group has a highly integrative task, such as process coordination or the design of an extremely interactive component (e.g., a data transmission component), it is often best to integrate it at a higher level.

Table II. Desirable Interface Characteristics

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 project's task, size, and stage. One size does not fit all. Each important interface deserves explicit, personalized, unique attention and optimization.

System teams must be built as carefully as IPTs. Like IPTs, they need charters. It may be necessary to provide team-building training to each system team's constituent members. As Sheard and Margolis [1995] found in their study of organization structures at Loral:

> It is very important to clarify the charters and limitations of all teams. Who will resolve inter-product-team issues? Who will make system architectural decisions? Who will hold cost accounts and work packages? Can managers override decisions made by teams, and if so, when, and by what process? How will decisions be communicated among these groups and to and from the product teams? Who is responsible for maintaining discipline processes? Will the product teams include the customer or will a system team do the interfacing?
>
> Projects [that] have answered these questions during team workshops report reduced confusion of roles and possibly reduced conflict as a result.

In addition, appropriate management and leadership roles for the system team and project must be determined. Accordingly, Mohrman et al. [1995: 133] add the following organization design questions:

> What management functions should the members of work teams or integrating teams perform, and what functions should people in managerial roles perform? How much self-management should be vested in teams? What leadership roles should be established in teams? Under what conditions should people without hierarchical authority (for example, team leaders) perform leadership functions?

A DFI process helps bring these questions and others to the forefront during the design of the project organization.

Grouping IPTs and FSGs into system teams is an essential step in a DFI process, for it determines the key interfaces in the organization. Table II lists the desirable characteristics of information transfer interfaces. The goal of this DFI step is to come up with the best system team groupings and the beginnings of a viable *scheme for interface mediation* (SIM) for the project. A SIM document explicitly outlines the expected interfaces, their desired characteristics, and the means of monitoring them. It is not given a formal outline here, although it should contain the relevant details of the tradeoffs involved in the design of the integration scheme as well as the implementation plan. As a formal document, however, it should not be an end in itself. The SIM document will serve the project's organizers as an invaluable interface management tool. Results of SIM development are included as part of the project's Systems Engineering Management Plan (SEMP),[5] Integrated Master Plan (IMP), and/or Program Execution Plan (PEP).[6] The quality of the system team and integration level determinations will directly affect a project's success.

3.4. Apply Integrative Mechanisms (IMs)

Integrative mechanisms (IMs) are strategies and tools for effectively coordinating actions between organizational components. IMs are catalysts, facilitating information flow across organizational, locational, cultural, traditional, and other barriers. IMs must regulate infor-

[5]For more details on a SEMP, see, for example, [Blanchard and Fabrycky, 1998: DoD, 1994; DSMC, 1990; Shishko et al., 1995].

[6]As described by [Peters, 1995: 3], a PEP was used for the organizational transition from a functional organization to IPTs in the Space Station project. "The PEP represents customer needs in terms of project strategies and issues, unique contract requirements, project requirements, project schedules and budget, unique and standard processes as well as IPT responsibilities and structure ... i.e. their operating mode, schedule development, resource allocation, organizational structure, process, and overall team authority and responsibilities."

mation flow such that it is available but not overwhelming. The categories of IMs listed in Table III represent the tools in an organizational integrator's "tool kit."

Once IPTs, FSGs, and system teams have been determined and interface characteristics have been established, organization designers should choose one or more appropriate IMs (based on the unique physical, political, architectural, and other attributes of the project) to facilitate each interaction. Browning [1998] provides insights on appropriate choices of IMs. Nadler and Tushman [1988] call this step "strategic linking." The deliberations in this step—especially the reasons for choosing particular IMs—should be recorded in the SIM document so they can be communicated and revisited (Step 6).

3.5. Manage Interfaces

After designing the best possible organizational structure and choosing appropriate IMs *a priori*, project management must maintain smooth information flow. This includes mediating technical issues via suitable IMs, monitoring the effectiveness of communication, and changing IMs when necessary. Evaluating the effectiveness of the SIM is difficult on a real-time basis. Often, only "lagging indicators" such as adherence to budget and schedule are available as macro level, proxy metrics. This is one reason why successful, sustained organizational integration is notoriously difficult. One proposed metric includes monitoring change notices resulting from interface issues. Integrated prototype testing and design reviews also provide opportunities to formally evaluate the effectiveness of organizational integration. Are the issues resulting 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.

Table III. Integrative Mechanisms[a]

1.	Improved information and communication technologies—collaborative tools, linked CAD/CAM/CAE systems, e-mail, tele- and videoconferencing, common databases (easily accessed and shared), common nomenclature, etc.
2.	Training—especially in team-building (and "system team-building" and "project-building"); raising awareness about integration needs and roles
3.	Co-location—physical adjacency of IPT, FSG, system team, and/or project members
4.	Traditional meetings—face-to-face gatherings for information sharing and/or decision-making
5.	"Town meetings"—not to share technical information, but to boost camaraderie and to increase awareness of project-wide issues
6.	Manager mediation—"up-over-down" (hierarchical) issue mediation schemes; heavyweight product managers 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

[a]Adapted from [Browning, 1998].

3.6. Reassess status

As the project evolves, the organization and the SIM should be reevaluated. Some interfaces will become more important, others less. New interfaces will form. Some will disappear altogether. Some IMs will no longer be appropriate. For example, IPTs will tend to call upon design support FSGs early in the design process; later, they will spend more time with production process FSGs. As organizational components and their interfaces grow and recede, the IMs must adapt or change. A group of organization designers should have the periodic task of reevaluating the SIM and effecting the necessary changes. Documenting this process well will greatly improve its long-term effectiveness, especially in lengthy development projects where turnover among the membership of the SIM group is likely.

4. IMPLEMENTATION BARRIERS

Despite the attractiveness of incorporating DFI considerations explicitly during organization design decisions, barriers exist. First, the performance gaps resulting from poor organizational integration must be acknowledged. Second, traditions incite resistance to organizational changes. Organization changes are not easy. New projects are the best opportunities to make necessary changes. This also argues for organization design foresight early in the project. Third, a suitable group of individuals to execute a DFI process may be hard to find and assemble. Obviously, they must have a systems perspective and the authority or support of authority necessary to enact their decisions. At a minimum, key project managers and potential system team managers should participate.

5. CONCLUSION

A DFI process provides a systematic approach to organization design that accounts for interteam integration issues. Such issues cause numerous complications and setbacks in complex system development projects. Designing organizations for integration results in improved information flow, better coordination, situation visibility, reduced rework, and less frustration for participants. DFI uses the interactions among components of the system architecture to understand likely interactions between activities in the development process, and both are in turn used to understand the likely interactions among groups in the organization. Once these interactions are identified, they can be managed via appropriate IMs. The organization design can be documented and reassessed over the life of a lengthy development project, although a DFI process certainly makes sense for shorter projects as well.

Organization design processes incorporating aspects of the proposed DFI framework are developing in several projects that recognize the importance of integration issues and their potential to influence project success. The DFI template presented here is generic enough to be suitable for a wide range of projects, yet powerful enough to provide explicit directions towards the anticipation and resolution of organizational integration issues. In the development of large, complex systems, a systematic approach is required to ensure an effective product, process, and organization. Companies can pursue the launch and management of complex projects as a "core competency" or capability. Companies that learn and begin improving upon a systematic process for organization design will be a step ahead.

ACKNOWLEDGMENTS

Massachusetts Institute of Technology's Lean Aerospace Initiative, sponsored jointly by the U.S. Air Force and a consortium of aerospace companies, provided direct support for this work. Additional support has come from MIT and a National Science Foundation graduate fellowship. The author is grateful for helpful comments on earlier versions of this paper from Stanley Weiss, Steven Eppinger, Charlie Boppe, and three anonymous reviewers. Previous versions of this paper include a conference paper [Browning, 1997] and an unpublished Master's thesis [Browning, 1996].

REFERENCES

C. Alexander, Notes on the synthesis of form, Harvard University Press, Cambridge, MA, 1964.

S.S. Altus, I.M. Kroo and P.J. Gage, A genetic algorithm for scheduling and decomposition of multidisciplinary design problems, Proc 21st ASME Des Automa Conf, Boston, September 1995.

C.Y. Baldwin and K.B. Clark, Design rules: The power of modularity, MIT Press, Cambridge, MA, 1999, in press.

B.S. Blanchard and W.J. Fabrycky, Systems engineering and analysis, 3rd ed. Prentice Hall, Englewood Cliffs, NJ, 1998.

T.R. Browning, Systematic IPT integration in lean development programs, Master's Thesis (Aero./TPP), Massachusetts Institute of Technology, Cambridge, MA, 1996.

T.R. Browning, Exploring integrative mechanisms with a view towards design for integration, Proc Fourth ISPE Int Conf Concurrent Eng. Res Appl, Rochester, MI, August 20–22, 1997, p. 83–90.

T.R. Browning, Integrative mechanisms for multiteam integration: Findings from five case studies, Sys Eng 1 (1998), 95–112.

W.E. Cole, Cooking up a batch of team synergy: Ingredients for setting up successful teams, Prog Manager (1995), Sept.-Oct., 28–33.

DoD, MIL-STD-499B Systems Engineering (Draft), DoD Military Standard Specifications MIL-STD-499B, May 6, 1994, Washington, DC.

DSMC, Systems engineering management guide, Defense Systems Management College, Fort Belvoir, VA, 1990.

S.D. Eppinger, A planning method for integration of large-scale engineering systems, Proc Int Conf Eng Des ICED—97, Tampere, Finland, August 19–21, 1997.

S.D. Eppinger, D.E. Whitney, R.P. Smith, and D.A. Gebala, A model-based method for organizing tasks in product development, Res Eng Des 6 (1994), 1–13.

C.I.G. Fernandez, Integration analysis of product architecture to support effective team co-location, Master's Thesis (ME), Massachusetts Institute of Technology, Cambridge, MA, 1998.

J.O. Grady, System integration, CRC Press, Boca Raton, FL, 1994.

R.K. Gulati and S.D. Eppinger, The coupling of product architecture and organizational structure decisions, Working Paper No. 151, MIT International Center for Research on the Management of Technology, Cambridge, MA, May, 1996.

J.R. Katzenbach and D.K. Smith, The wisdom of teams: Creating the high-performance organization, Harvard Business School Press, Boston, 1993.

J.A. Klein and G.I. Susman, Lean aircraft initiative organization & human resources (O&HR) survey feedback–integrated product teams (IPTs), White Paper LEAN 95-03, MIT Lean Aircraft Initiative, Cambridge, MA, 1995.

A. Kusiak and J. Wang, Decomposition of the design process, J Mech Des 125 (1993), 67–695.

K.R. McCord and S.D. Eppinger, Managing the integration problem in concurrent engineering, Working Paper No. 3594, MIT Sloan School of Management, Cambridge, MA, 1993.

N.F. Michelena and P.Y. Papalambros, Optimal model-based decomposition of powertrain system design, Proc ASME Adv Des Automat, 1995.

S.A. Mohrman, S.G. Cohen, and A.M. Mohrman, Jr., Designing team-based organizations, Jossey-Bass, San Francisco, 1995.

M.D. Morelli, S.D. Eppinger, and R.K. Gulati, Predicting technical communication in product development organizations, IEEE Trans Eng Manage 42 (1996), 215–222.

D.A. Nadler and M.L. Tushman, Strategic organization design: Concepts, tools, and processes, Scott, Foresman, Glenview, IL, 1988.

J.F. Peters, The transition of functional organizations to integrated product teams on the space station program, Proc Fifth Annu Int Symp INCOSE, St. Louis, July 22–26, 1995.

T.U. Pimmler and S.D. Eppinger, Integration analysis of product decompositions, Proc ASME Sixth Int Conf Des Theory Methodol, Minneapolis, MN, September, 1994.

E. Rechtin, Systems architecting: Creating & building complex systems, Prentice Hall, Englewood Cliffs, NJ, 1991.

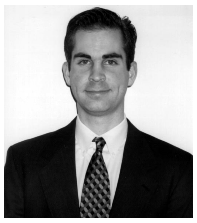
D.G. Reinertsen, Managing the design factory: A product developer's toolkit, The Free Press, New York, 1997.

R. Sanchez and J.T. Mahoney, Modularity, flexibility, and knowledge management in product and organization design, IEEE Eng Manage Rev (1997), 50–61.

S.A. Sheard and M.E. Margolis, Team structures for systems engineering in an IPT environment, Proc Fifth Annu Int Symp INCOSE, St. Louis, July 22–26, 1995.

R. Shishko et al., NASA systems engineering handbook, SP-6105, NASA, 1995.

P.G. Smith and D.G. Reinertsen, Developing products in half the time, Van Nostrand Reinhold, New York, 1991.



Tyson Browning conducts applied research and provides internal consulting on engineering process development for Lockheed Martin Tactical Aircraft Systems in Fort Worth, Texas, USA. He previously worked with the Product Development Focus Team of the Lean Aerospace Initiative (http://lean.mit.edu/public) at Massachusetts Institute of Technology. His education includes: B.S. In Engineering Physics from Abilene Christian University; S.M. in Aeronautics and Astronautics from MIT; S.M. in Technology and Policy from MIT; Ph.D. in Technology, Management, and Policy (interdisciplinary technology management and systems engineering) from MIT.