Proceedings of the Sixth Annual International Symposium of INCOSE, Boston, Jul. 7-11, 1996, pp. 787-794.

MULTI-TEAM INTEGRATION: INTERDEPENDENCE AND INTEGRATIVE MECHANISMS

Tyson R. Browning
Massachusetts Institute of Technology
77 Massachusetts Avenue
Room 33-407
Cambridge, MA 02139
tyson@mit.edu

Abstract. The drive towards Integrated Product Development (IPD) includes an impetus to organize around Integrated Product Teams (IPTs). The use of IPTs has brought with it many issues, including those at the IPT interfaces. Program integration (cross-functional, upstream/downstream, customer and supplier) can exist at several levels, within IPTs and between IPTs. This paper focuses on the realm of IPT interdependence and categorizes several Integrative Mechanisms (IMs) to facilitate interteam integration. IMs are strategies and tools for effectively coordinating actions across IPTs within a program. This paper is based on studies in the aerospace industry, but the implications extend to any large, complex development program.

TEAM INTERDEPENDENCE

Product design potentially involves hundreds if not thousands of individuals who make millions of design decisions over several years. Few of these many determinations can be done in isolation: design choices involve tradeoffs which affect many other design, process, cost, and operational parameters. In support of this interdependence, product design managers have as an essential task the facilitation of the transfer of information among design groups. (Allen, 1977) "Their 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) Therefore, there exist further levels of integration within each program: not only must the product teams be integrated as Integrated Product Teams (IPTs),[1] but also the IPTs themselves must be integrated on a macro level. Determining these higher levels of integration is a systems engineering issue and the focus of this paper.

A product and a program are usually designed intuitively, hopefully informed by a systems engineering perspective. Major subsystems composed of many elements are identified and allocated to organizational entities. Within these groupings, IPTs are formed to develop the various elements. Hence, more complex subsystems generally have multiple IPTs involved in the development of their elements. Task sizes must match IPT capabilities. Task allocation to IPTs should consider task overlap and "underlap." Tasks should not be redundant (unless by design), nor should they "fall between the cracks" so that each IPT thinks that "someone else" is handling them. Furthermore, task sequencing becomes challenging when once serial tasks are now performed in parallel—the implementation of concurrent engineering—by IPTs. Thus, integration at the IPT level requires integration on subsystem and system organizational levels. Special concern should be given to the information that will flow between IPTs as they carry out their concurrent engineering tasks.

Such a program design process gets increasingly difficult as system complexity increases. (Complexity here implies numerous, highly-coupled subsystems.) The IPTs which develop such systems 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 team C is using for parameters w and z; team C needs to know the result of team A's activities to determine w and z. Such couplings may imply a slow, iterative development process. The amount of coupling and the number of iterations increase exponentially with system complexity. As the level of task and IPT

[1] IPT integration involves bringing in representatives from the breadth of the disciplines involved throughout the life cycle, thus including multiple functions and representatives from both upstream and downstream processes.

interdependence increases, the traffic on inter-IPT communication channels increases.

Additionally, more complex systems generally imply a greater number of IPTs. Figure 1 (Browning, 1996) shows how the number of inter-IPT communication channels increases with the number of IPTs. (Of course, not every IPT will have to interface with every other IPT.) If IPTs are truly held to their useful size—i.e., 10-15 persons—then their number will most certainly be large for complex programs. Today, many programs have questionably formed "IPTs" with 70, 100, or more members. As programs move towards effective implementation of the IPT paradigm, the number of IPTs generally grows and the issue of IPT integration becomes more acute.

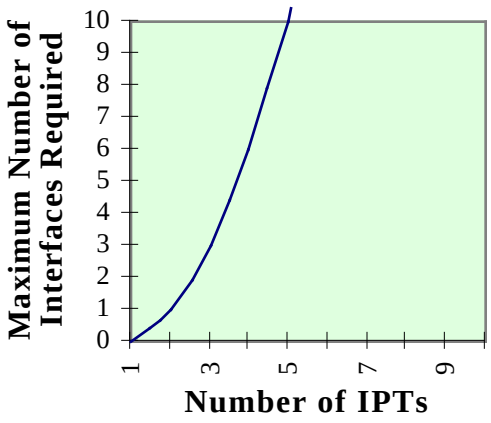

Figure 1: Number of Inter-IPT Interfaces Increases with Number of IPTs

Given these challenges in complex programs, one perceives the potential for issues in IPT integration. In the automotive industry, integration difficulties have been explicitly noted: "One shortcoming that became apparent as multi-functional teams [IPTs] worked on a specific product was lack of coordination within the Product Engineering group. This lack of coordination resulted in interface problems between Simultaneous Engineering teams [again, IPTs] that should have been solved functionally within Product Engineering." (Mattis, 1992; notice to IPTs added) (McCord and Eppinger, 1993) also discerns the potential for communication problems between interdependent teams:

> Relying solely on … an informal communication network for integration … means to depend on the engineers to comprehend and initiate all of the necessary interactions between PDTs [Product Development Teams]. Unfortunately, engineers are rarely sensitive to all inter-PDT relationships, especially concerning how their work affects the work of other PDTs. Furthermore, many unforeseen conflicts between PDTs arise throughout the course of the project which are too slowly resolved through informal integration. … More formal, planned integration mechanisms must be designed into the organization to ensure necessary information exchange between PDTs and to expose and resolve inter-PDT issues as early and as quickly as possible.

INTERFACE MANAGEMENT

Aware of the issues, how should one approach them? First, program design should have as a goal the optimal number of IPTs assigned to the correct, well-defined tasks. This is a systems engineering issue and follows directly from the architectural breakdown of the system. Models of component interrelationships and anticipated information flows can facilitate this determination. Although the system perspective can be advantageous to a program at any time, it is especially essential as an *a priori* consideration, as a program is being designed (i.e., designing the program for integration). Second, the IPTs must be integrated in such a way that the interfaces between them allow for information transfer in the optimal fashion. This requires the use of Integrative Mechanisms (IMs). IMs should be considered during program design, but also should lend themselves to *a posteriori* application to existing programs. With all of these goals, one must realize that "No complex system [or organization] can be optimum to all parties concerned nor all functions optimized." (Rechtin, 1991) But some approaches will achieve better results than others.

What would desirable interfaces look like? To what ends are the means applied? Table 1 (Browning, 1996) notes some characteristics interfaces should strive to exhibit.

INTEGRATIVE MECHANISMS

Integrative mechanisms (IMs) are strategies and tools for effectively coordinating actions across groups (IPTs and/or functional support groups, etc.) within a program. As catalysts, they facilitate information flow across communication barriers, such as a company or program's organizational structure, incentive systems, location, leadership styles, cultural differences, and management traditions. (Morelli, 1993) They must also regulate information flow. They may be thought of as the tools in an integration "tool kit." Each IM's appropriateness may not hold for every program or interface: one must consider a firm's organizational environment (culture and personalities) and a program's

technical information requirements when applying any of these approaches.

Interfaces should be…

- **tight-fitting**, in terms of task assignment. Tasks should not overlap or "underlap."
- **permeable**, in terms of information flow. Information must arrive "just-in-time"—not too early or too late. It must be the needed information—not more and not less. It must flow readily and smoothly, yet not inundate its recipients.[2] The interface must allow just the right amount of the right information to flow.
- **defined**, in terms of what information needs to flow and where and when (i.e., "the *right information* at the right place at the right time").
- **direct**, in terms of distance from provider A (IPT or IPT member) to recipient B (IPT or IPT member). This path should be free of undue bureaucracy or other delays.
- **manageable**, in terms of regulating information flow. The interface must have a means of altering what information gets transferred, when, and how.
- **recordable**, in terms of allowing documentation of information flow. Information useful once may be useful again. To avoid "reinventing the wheel," one needs a record of the flow.
- **verifiable**, in terms of allowing analysis of success and flow rate. Success must be objective, not subjective (inasmuch as is possible).
- **adapted**, in terms of the program's task, size, and stage. One size does not fit all. Each interface (or at least each type) deserves explicit, personalized, unique attention and optimization.

Table 1: Desirable Characteristics of IPT Interfaces

Effective utilization of IMs requires understanding of their definitions and limits as one determines the applicability of each to a given program's task, size, and phase. It is beneficial to note that "it is easier to match a system to the human one it supports than the reverse." (Rechtin, 1991) Furthermore, the use of IMs often invites a tradeoff. For example, improved information and communication technologies can be traded with co-location to an extent. The varying content of any arbitration or management group can also be balanced between pros and cons at each end of the spectrum. The following sections describe nine IMs: (1) Systems engineering (and modeling), (2) Improved information and communication technologies, (3) Manager arbitration, (4) Participant arbitration, (5) Interface "management" groups and integration teams, (6) Co-location, (7) Interface minimization, (8) "Town meetings," and (9) Training.

SYSTEMS ENGINEERING (AND MODELING)

As an IM, systems engineering is ideally an *a priori* technique, informing the IPT breakout based upon the requirements and specifications of product architecture. Even after a program gets underway, interface management continues through explicit interteam checks on specific parameters, etc. Perhaps systems engineering should not be classified as an IM in the strictest sense: it might be better thought of as the work gloves one must put on before using the other IM tools.

It is important to note the role of software tools and modeling techniques in facilitating the systems engineering approach. Requirements documentation, flowdown, and analysis tools all play a crucial role. Furthermore, modeling techniques such as the Design Structure Matrix (DSM)[3] can inform the systems engineer as to the necessity and appropriate use of other IMs. Modeling promises to become even more important in the future, as "one by one, government agencies and engineering companies … [replace] requirements for documents with requirements for models." (Scruggs, 1994)

IMPROVED INFORMATION AND COMMUNICATION TECHNOLOGIES

Information and communication technologies serve to penetrate communication barriers and to increase the capacity, efficiency, and general efficacy of a program's information exchange.[4] Naturally, the possibilities within this broad category are many. Table 2 (Browning, 1996) lists several representative technologies in this group.

Improving shared databases implies embellishing the breadth and depth of the data stored, increasing accessibility while decreasing access time, and training an ever-greater amount of the workforce in their use.

[2] Picture receiving several hundred program-related e-mails each day, for instance. The mere presence and availability of information is not enough! One can become desensitized…

[3] For a summary of DSM applications, see (Browning, 1996).

[4] For specific research on these intentions, consult (Hauptman and Allen, 1987) and (Jakiela and Orlikowski, 1990).

Ideally, an IPT member could access all databases easily and routinely from a single, common-place terminal, such as a personal computer on their desk. Engineering, manufacturing, cost, test, and other data of many types should be so archived.

- linked CAD tools
- shared databases
- e-mail
- voice mail
- standardized hardware and software suites
- local and wide area networks (LANs and WANs)
- archival databases for mail and meeting minutes (with friendly search and retrieval mechanisms)
- computer and/or video conferencing
- teleconferencing

Table 2: Some Integrative Information and Communications Technologies

Note that this IM includes several information tasks: transfer (dissemination), access, and assimilation. Technologies facilitating any one of these areas may not necessarily further them all. For example, some technologies, such as teleconferencing, make information exchange so expedient that the propensity to not document that exchange increases. One must consider such factors if record keeping is a priority.

Much more has been said in other places about the roles many of these technologies play *within* IPTs—e.g., the importance of CAD/CAM to concurrent engineering.[5]

MANAGER ARBITRATION

This IM comes in (at least) two flavors, both of which have in common the facilitation of interface issues primarily by managers.

"Up-over-down." Some organizations use an "up-over-down" approach to interface management as shown in Figure 2 (Browning, 1996). Managers above the IPTs arbitrate interteam issues rather than having different teams' members deal with them directly. This mechanism works better for relatively independent teams requiring little information transfer and the resolution of few issues. As the complex coupling between teams increases, however, management quickly becomes overloaded and a barrier to information flow. Note that in this arrangement management is generally reactive as issues arise, stepping in only to resolve issues or review progress.

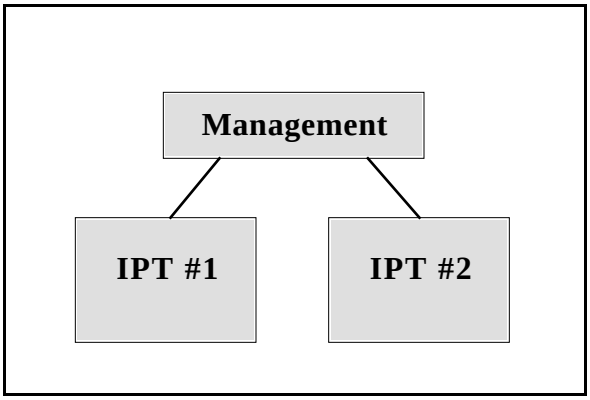

Figure 2: "Up-over-down" Manager Arbitration

Heavyweight Product Managers (HPMs, or Integrators). A HPM, as defined by Clark and Fujimoto, has "direct access to the working-level engineers" when necessary and exercises "strong direct and indirect influence across all functions and activities in the project." (Clark and Fujimoto, 1991) The HPM has more clout than the functional managers. Table 3 lists the characteristics of successful HPMs in the automotive industry, as compiled by (Clark and Fujimoto, 1991). After reviewing this list, one should discern the difficulty in finding such a superhuman individual, especially in the aerospace industry, where complexity and scope are much greater. Perhaps the key here is to contain these characteristics in a cadre of individuals who perform the HPM role together (with one individual as this group's leader).

(Lawrence and Lorsch, 1967) describes this role as that of an "integrator." This person must possess status, expertise, and personal attributes consistent with the IPTs' ideals. HPMs/Integrators are proactive, anticipating interteam issues. Clear product vision and influence dwelling in a single individual make HPMs effective integrators in some circumstances. It is rare, however, to find all of the necessary attributes in one individual for large, complex development projects such as those connected with system integration in the aerospace industry.

While this role/IM focuses on the individual's (or small group's) integrative influence, it does not exclude direct IPT interfaces, as does the up-over-down approach. On the contrary—it encourages them.

[5] See (Robertson and Allen, 1990 and 1991) and (Murotake, 1990).

- Coordination responsibility in wide areas, including production and sales as well as engineering
- Coordination responsibility for the entire project period from concept to market
- Responsibility for concept creation and championing as well as cross-functional coordination
- Responsibility for specification, cost target, layout, and major component choices
- Responsibility for ensuring that the product concept is accurately translated into technical details
- Frequent and direct communication with designers and engineers at the working level as well as through liaisons
- Maintain direct contact with customers
- Possess multilingual and multidisciplined abilities in order to communicate effectively with marketers, designers, engineers, testers, plant managers, controllers, and so forth
- Role and talents in managing conflict surpass those of neutral referees or passive conflict managers; they may initiate conflicts to prevent product designs or plans from deviating from the original product concept
- Possess market imagination and the ability to forecast future customer expectations based on ambiguous and equivocal clues in the present market
- Circulate among project people and strongly advocate the product concept rather than do paperwork and conduct formal meetings
- Mostly engineers by training, they possess broad (if not deep) knowledge of total vehicle engineering and process engineering

Table 3: Characteristics of Effective Heavyweight Product Managers in the Auto Industry[6]

PARTICIPANT ARBITRATION

This IM serves as a catch-all for several categories of non-management interface arbiters. Most of the terminology for these subcategories comes from the automobile industry. (McCord and Eppinger, 1993)

Conflict Resolution Engineers (CREs). When technical conflicts are brought to their attention, CREs act as dedicated arbitrators between IPTs. "Conflict" refers to disagreements over technical issues or trades that affect two or more teams. An example of a CRE is a "zone engineer," who arbitrates technical conflicts between teams within a given section of a program. The CRE handles "turf" issues of a technical nature, provided that "both sets of turf" fall under his or her jurisdiction. If the CRE comes from the ranks of management, the role would fit under Management Arbitration. More likely, however, the CRE would be a functional guru from the technical area within question.

Liaisons. Akin to a CRE, a liaison also works to resolve technical issues at team interfaces. However, liaisons play a more proactive role as they facilitate continuous and intensive information exchange. This role involves seeking out technical conflicts, discovering them earlier, and resolving them faster by participating in interteam interactions. A liaison is a member of one IPT and serves that team entirely (as far as the liaison role is concerned) by facilitating that IPT's communications with all other IPTs—a "from one to many" and a "from many to one" relationship.

Engineering Liaisons (ELs). Whereas liaisons reside in one IPT, ELs are formal members of two or more IPTs whose interface(s) the EL coordinates. Hence, the EL will go to the team meetings and other team activities of two or more IPTs. Not only do ELs establish and maintain a firm communication link between IPTs, they also perform specific technical tasks on at least one of the teams. They are working, development engineers.

INTERFACE "MANAGEMENT" GROUPS AND INTEGRATION TEAMS

This IM includes groups, perhaps a mixture of both management and participants, that manage IPT interfaces, largely from an "up-over-down" perspective. (Here "management" is enclosed in quotations as referring to management of the interface itself, which may or may not be done entirely [if at all] by managers. If such a group is composed entirely of management personnel, it should fit more properly under Management Arbitration.) Note the difference between *integrated* teams and *integration* teams, which may or may not themselves be "integrated" (i.e., composed of cross-functional, upstream/downstream, supplier, and customer representatives). At least two genres of such "integration teams" exist: predetermined and impromptu (for lack of better terms).

[6] Source: (Clark and Fujimoto, 1991).

Predetermined. Predetermined integration teams are formed from the outset of a program (or their date of formation is fixed from the outset), set up to deal with foreseen issues of a complex, critical nature that involve multiple IPTs. Such preordination would most likely stem from a systems engineering analysis that reveals the crucial interfaces based on requirements and specifications.

Impromptu. Impromptu interface management groups also deal with a single, complex, critical issue concerning multiple IPTs. However, these teams are formed when such an issue crops up in the middle of an ongoing program, unforeseen by the systems planners (at least at the program's outset). They dissolve when the issue is resolved. Often, such groups form to handle action items from product reviews. Impromptu integration teams consist of engineers and others drawn from across a project. Essentially, they correspond to task forces, *ad hoc* teams, splinter teams, tiger teams, and action teams.

CO-LOCATION

Co-location involves positioning the IPTs and functional support groups within a program (already assuming co-location of the IPTs themselves) in close proximity (usually within sight and sound of each other, but sometimes within walking distance) for the purpose of facilitating communication, both formally and informally. Co-location offers at least two advantages: (1) it empowers issue resolution at the lowest levels; (2) it increases cross-functional appreciation. It is useful to think of co-location in terms of its influence on communication patterns rather than in terms of distance. If "co-located" IPTs or groups still use the phone as the primary means of communication, for example, perhaps they have not tapped the true advantage. If individuals still reside in mazes of cubicles, where seeing if another person is at their desk requires leaving one's own desk—again, perhaps a key benefit of co-location is yet to be realized. The extent to which co-location adds value (i.e., how far to take it) is still a subject for additional research. (Undoubtedly, like other IMs, it will vary by program task, size, and stage.) Finally, co-location offers one of the most obvious examples of how IMs can be traded off against each other. Greater co-location can reduce the need for other IMs.

INTERFACE MINIMIZATION

This IM is actually more of a preventative measure—attempting to make IPTs as technically independent as possible, based on the premise that minimal interfaces imply fewer interface issues. Of course, this ultimately stems from the product's architecture—the more modular the architecture, the more independent the teams. At the organizational level, IPT independence can be facilitated by modeling the required amount of information flow. As (Rechtin, 1991) notes, "Choosing the appropriate aggregation of functions is critical in the design of systems." Likewise, choosing the appropriate aggregation and integration of organizational functions becomes the critical task of the organizational designer.

Choosing how to break down a program into IPTs can benefit from some of the same heuristics (Rechtin, 1991) has collected in reference to breaking down a system's architecture. Four such heuristics are collected in Table 4. Here, "communications" refers to "interrelationships, connections, interplay, information flow, etc." Thus, both subsystems and IPTs should be semi-isolated to a reasonable extent so that a minimal number of external events have the potential to disturb the inner workings. (Rechtin, 1991) sums up this concept with a heuristic akin to that of minimum communications: "Design the elements to make their performance as insensitive to unknown or uncontrollable external influences as practical." In a sense, this is also appropriate for IPTs.

- In partitioning, choose the elements so that they are as independent as possible—i.e., elements with low external complexity and high internal complexity.[7]
- In partitioning a distributed system, choose a configuration in which local activity is high speed and global activity is slow change.[8]
- In partitioning a system into subsystems, choose a configuration with minimal communications between the subsystems.[9]
- Do not partition by slicing through regions where high rates of information exchange are required (e.g., computers).

Table 4: Partitioning Heuristics

[7] From Alexander, Christopher, *Notes on the Synthesis of Form.* Harvard University Press, Cambridge, Mass., 1964. Qtd. in Rechtin.

[8] From Courtois, P.J., "On Time and Space Decomposition of Complex Structures." *Communications of the ACM* 28, 6: 590-603, June 1985. Qtd. in Rechtin.

[9] Excepting massively parallel neural networks

Program integration can be approached at three levels: 1) IPTs, 2) "system teams," and 3) program. System teams are collections of IPTs (and perhaps functional groups) with especially strong interdependencies. Together with other system teams and disparate IPTs, they form the program. To some extent, the three levels can be traded off against each other. For example, in a large program not all functions can be represented on every IPT at the lowest level. Not all parties can be co-located due to certain constraints. Less integration at the first level (within the IPTs) will require much more proactive integration (i.e., application of IMs) at the second and third levels. Conversely, well integrated and independent IPTs will require fewer IMs at the second and third levels. Interface minimization seeks to push many of the barriers and issues found at the second and third levels down to the first. While technically independent IPTs have fewer issues to arbitrate with other IPTs, however, they usually have more to resolve within themselves. Keeping IPTs to an effective size and limited resources trade with IPT independence and interface minimization.

"TOWN MEETINGS"

Town meetings (and other similar rallies) gather everyone on a program together in one place to review the program's progress. Although relatively ineffective for transferring technical information, they serve to boost morale and camaraderie. On large programs, especially ones that span several facilities in several states, such gatherings are logistically impossible. Smaller versions (e.g., all employees at a given site) and permutations of meetings with these goals also fit into this category.

TRAINING

Level one integration (i.e., at the IPT level) recognizes the necessity of training to boost team performance and sustain the competence of its members. In much the same ways as IPTs learn to operate as better teams, they can receive training on how to more adeptly arbitrate issues with other IPTs and utilize the IMs facilitating those interactions. Of course, training must be part of a larger program of improvement. And it must be available to the right people, at the right place, and at the right time—with the realization that failing these ideals quickly diminishes the value added.

CONCLUSION

These nine IMs are broadly representative of approaches taken within several industries to the mandatory task of IPT integration. Other categories and subcategories may exist as well. Application of IMs is clearly related to the character of the industry and the history of organizations within. Fundamental to the application of IMs is explicit recognition and consideration of the need for such integration at the time IPTs are established. IMs are most effective when applied appropriately—i.e., with a knowledge of their strengths and weaknesses in the environment under consideration and hopefully based on a systematic approach, stemming from the architectural structure of the product. These exhortations are further explored in (Browning, 1996).

ACKNOWLEDGEMENT

This paper is based on research funded by the Lean Aircraft Initiative at the Massachusetts Institute of Technology and the National Science Foundation.

REFERENCES

Allen, Thomas J., *Managing the Flow of Technology: Technology Transfer and the Dissemination of Technological Information Within the R&D Organization.* M.I.T. Press, Cambridge, Mass., 1977, 1984.

Browning, Tyson R. *Systematic IPT Integration in Lean Development Programs.* M.I.T. Master's Thesis (Aero./T.P.P.), June, 1996.

Clark, Kim B. and Takahiro Fujimoto. *Product Development Performance: Strategy, Organization, and Management in the World Auto Industry*. Harvard Business School Press, Boston, Mass., 1991.

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

Hauptman, Oscar and Thomas J. Allen. "The Influence of Communication Technologies on Organizational Structure: A Conceptual Model for Future Research." *Working Paper, M.I.T. Sloan School of Management,* 1987.

Jakiela, Mark John and Wanda J. Orlikowski. "Back to the Drawing Board? Computer-mediated Communication Tools for Engineers." *Working Paper, M.I.T. Sloan School of Management*, 1990.

Lawrence, Paul R. and Jay W. Lorsch. *Organization and Environment: Managing Differentiation and Integration.* Harvard Business School Press, Boston, Mass., 1967, 1986.

Mattis, David P. *Case Study on the Strategies of Systems Engineering.* M.I.T. Master's Thesis (Mgt.), June 1992.

McCord, Kent R. and Steven D. Eppinger. "Managing the Integration Problem in Concurrent Engineering." *Working Paper, M.I.T. Sloan School of Management*, 1993.

Morelli, Mark D. *Evaluating Information Transfer in Product Development.* M.I.T. Master's Thesis (Mgt.), June 1993.

Murotake, D.K. *A Double-Edged Sword: Relationships Between the Engineering Use of Computer Tools and Project Performance.* M.I.T. Ph.D. Thesis (Mgt.), 1990.

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

Robertson, David and Thomas J. Allen. "CAD System Use and Engineering Performance in Mechanical Design." *Working Paper, International Center for Research on the Management of Technology, M.I.T. Sloan School of Management*, 1991.

Scruggs, Johnny V. *Meta-Modeling for Project Engineering.* Arizona State University Master's Thesis, December 1994.

BIOGRAPHICAL NOTE

Mr. Browning is currently a masters student in the Department of Aeronautics and Astronautics and the Technology and Policy Program at the Massachusetts Institute of Technology. He will soon begin Ph.D. studies and research in the Technology Management and Policy Program at M.I.T. While in school, he has worked with the Lean Aircraft Initiative at M.I.T. for three years. Mr. Browning has previous work experience at Honeywell, Inc. and at Los Alamos National Laboratory.