Calvin Pava

Nonroutine Office Work[1]

Predominantly nonroutine office work often defies traditional socio-technical analysis. Multiple, concurrent, nonlinear conversion processes and professional separation render this approach inapplicable. This paper explores a socio-technical intervention based on the modified procedure. The setting is a software engineering group in a moderate-sized computer systems firm. At the time of the design, the department employed 52 professionals and 12 support staff. The steps taken were as follows:

Step O: Mapping the Target System

Step 1: Entry, Sanction, Start-up

Step 2: Initial Scan

Identify the Environment

Summarize Major Historical, Social and Physical Features

Formulate the Mission

Formulate the Philosophy

Step 3: Technical Analysis

List and Assign Priorities to Deliberations

Identify Different Forums

Identify Parties to Each Deliberation

---

[1]A revision of Chapter 5 in C.H.
y Pava, Managing New Office Technology: An Organizational Strategy. New York/London: Free Press/Collier Macmillan, 1983.

Step O: Mapping the Target System

The initial request for organization redesign came from a program team leader in a unit assigned to a large, innovative software development project for a highly advanced system the company was working on. He was concerned that the existing social and technical infrastructure, adequate for smaller projects, would prove insufficient to manage such a large undertaking, and he explored this situation with a consultant.

From the start it was apparent that changing only the organization of a few program team leaders reporting to this single development program leader would not generate

much improvement. Instead, a broader change initiative was necessary. At a minimum, the reconfiguration of work would have to embrace the entire software group involved in the computer system development project, perhaps spilling over into hardware engineering. Ultimately, organizing deliberations[2] that spanned all units working on the new computer would result in the trade-offs necessary for its success.

Step 1: Entry, Sanction, Start-up

The program team leader worked with the consultant to create interest in the proposed design effort. Discussions and instructional materials (cases of redesign in other organizations) eventually persua--ded all development program leaders and senior programmers to sanction redesign of the entire software group.

Representatives from different levels and specialities in the software group formed the design team. The team's mandate was to analyze the software group and to propose a more effective social and technical configuration. The team members were also asked briefly to analyze other groups involved in the computer system project and to suggest changes that might be made in their work system.

The development programmer convinced initially skeptical, reluctant top managers to champion the design effort. To do this, the heads of other units involved in the development of the computer system were invited to join a steering committee. This committee would periodically review the design team's work, providing a sounding board and perhaps

[2]Pava's "deliberations" and "discretionary coalitions" are detailed by Trist in "Commentary on Pava's Managing New Office Technology" in this volume.

smoothing the way for redesign of other units participating in the computer development project.

Because of severe time pressure on all team members, it was agreed in advance that the consultant would perform certain tasks on behalf of the design team; precisely what was initially left open. It was also agreed that the team was to meet once each week for a maximum of five hours and that the steering committee would meet with the design team on a monthly basis, with short updates written weekly by a design team member and the consultant.

Step 2: Initial Scan

The initial scan gets the design team to paint the "big picture"--how the system's software engineering organization has developed and how it currently functions.

Identify the Environment

The mission of a work system must be executed in relation to a dynamic environment. Socio-technical theory distinguishes two levels of the environment: the transactional and the contextual. The design team outlined the software group's transactional environment to include hardware engineering, marketing, customer service and divisional organizations. The contextual environment included an increasingly competitive labor market for software engineers; growing customer sophistication in purchase decisions; tight money, with resulting decline in brand loyalty; and stronger competition from innovative software companies. The environmental scan made the design team members more sensitive to the external forces that their work system had to contend with. Specifically, technical elegance alone was no longer

sufficient for success. Cost, timely delivery, complementary products and attractiveness to new talent were becoming greater factors in success.

Summarize Major Historical, Social and Physical Features

The design team recognized that the software group and other groups involved in the computer system project were too dispersed to permit easy coordination of work. Moreover, the project groups had grown so quickly that there was distrust and misunderstanding between old-timers, who were familiar with each other, and new people, who were not yet considered proven. Finally, the software group had shown a strong tendency to delay "debugging" in past projects. This had led to both premature product release, which exported problems to customers, and delayed product rollout. These outcomes served to generate resentments from the marketing and service organizations.

The initial scan compels design team members to begin viewing their work as a whole, rather than immediately focusing on specific problems or solutions. Also, during these early proceedings, team members begin learning the group dynamics they must master to function effectively as a team. During the initial scan, the software group design team settled upon ground rules for conducting its work. Finally, the scan made historical and other trends vivid, opening up areas of discussion hitherto closed and encouraging people to reflect and take action jointly on phenomena over which they had previously exercized little control. This capacity to identify and manage institutional issues was an outcome of the initial scan, which endured longer than any particular substantive finding.

Formulate the Mission

The design team defined the group's mission: to provide advanced system functions and reliable programs that served customer needs at minimum economic and social cost. Mission statements sound glib to some but represent careful consideration. Team members linked specific top-priority objectives in their group's mission (such as advanced functions, reliable programs, customer service and minimum costs) to key elements of the company's strategy. Commitment to providing advanced system functions arose from the advanced nature of the software project, which was intended to deliver a higher level of cost performance than earlier products. Concern for reliabilty was based on earlier experiences with hasty debugging that had exported software problems to customers and impeded rapid buildup of market share. Serving customer needs was a priority rooted in the firm's renowned orientation to market responsiveness (especially adopting popular features before other firms). Minimum costs were important in light of the firm's tight capital and the need to keep good people from burning out. Emphasis upon programs rather than code was particularly meaningful; it represented a fundamental reconceptualization of the software group's business. Typically, priority had been given to code, or individual lines of commands written into a program. "... good lines of code per day" was a standard way of assessing performance in the software group. But the industry trend was that software products were becoming much more complex. As this shift occurred, interrelations between subsections of a program became more critical. To do a good job, system analysts and programmers had to be cognizant of much more than the subsections of code that they were immediately responsible for composing. Reference to a bigger picture was vital, especially in the later stages, as correction of any one part of the program required alteration of

other modules. The design team decided to shift everyone's focus by designating good overall programs as the mission of the unit's efforts. The mission statement was reviewed with the steering committee, which proceeded to sanction it.

Formulate the Philosophy

The design team articulated a philosophy about the way people should be managed in the software group:

> • With sufficient understanding and skills, employees are professionally suited to a high degree of self-direction within a framework of mutual honesty and mutual constraint.
>
> • Not all employees seek identical career development. Both specialist and generalist talent are needed by the software group. Multiple paths of development are therefore necessary to create a variety of alternatives. Such development must be supported by individual initiative, joint discussions and periodic formal appraisals.

Like the mission statement, this philosophy may initially sound glib. Yet interpreted in the context of the software unit's history, it diplomatically summarized important value choices. The first clause, calling for professional self-direction with mutual honesty and constraint, was an attempt to strike a firm balance between extreme authoritarianism and participative management, styles between which the group had earlier oscillated. The second clause, referring to career development, sought to legitimize a rising need for different talents as the business grew, a

situation that previously had led to petty status distinctions and arbitrary promotion decisions. Hence, the philosophy statement involved human resource issues that were vital to the group's continued success.

Step 3: Technical Analysis

Nonroutine office work predominantly involves multiple, often nonlinear, conversion processes. This part of the analysis proceeds by mapping deliberations; the traditional variance matrix is used only for strictly routine tasks.

List and Assign Priorities to Deliberations

Technical analysis of nonroutine office work begins with listing all deliberations in which managers and professionals take part. This is done by having the design team specify what topics must be settled for the unit's mission to be fulfilled. Accordingly, the design team identified over 35 deliberations that went on within the software engineering group and between this group and its environment, including:

- stipulation of system features;
- outline of system documentation;
- PERT (Program Evaluation and Review Technique) timeline development and management;
- test/target machine allocations;
- employee development and advancement;

• declaration of different design editions;

• keeping pace with hardware changes;

adjudicating model debugging with integrity of initial system architecture;

reconciling late debugging compromises with initial system specifications.

It took the team some time to decide what constituted a deliberation. Then alternative formulations were tried in order to produce the right balance of parsimony and detail. After the list was settled on, the team rank-ordered the items to ensure that the more important deliberations would be fully analyzed. Less vital deliberations would be analyzed partially or on an as-needed basis. The team chose thus to analyze 18 of the more than 35 rank-ordered deliberations.

Identify Different Forums

At this point, the design team was asked to identify the different forums in which each major topic was to be deliberated and to classify these forums according to their level of formality (structured, semistructured and unstructured). A sample of the analysis appears in Table 1.

Identify Parties to Each Deliberation

Next, a design team will specify who is involved in each important deliberation. First, a list of current participants is drawn up. Second, information taken from the deliberation and information contributed to it are noted for each participant. Third, a revised list noting who

Table 1

Forums Associated with a Major Deliberation:

Stipulate Systems Features

Type of forum	*Topic*
Structured	Annual strategic planning cycle
	Industry reports and periodicals
	Quarterly marketing reports
	Beta test site conference and user conference
Semi-structured	Conventions and conferences
	Post-project reviews
	Division and corporate technical seminars
Unstructured	Troubleshooting customer problems
	Ad hoc technical discussions and exchanges

ideally should be party to the deliberation is posted, excluding current participants who do not belong and including overlooked parties. Finally, the ideal list is annotated to indicate what information each party brings to the deliberation. These data are set aside for use during the social analysis, which follows the technical analysis. The software design team used the procedure outlined here, and a portion of this analysis is summarized in Table 2.

Table 2

Parties Ideally Associated with a Major Deliberation: Declare Different Design Editions

Ideal participants	*Information taken from deliberation*	*Information contributed to deliberation*
Senior programmers	Progress reports and reviews Interface with hardware engineers on pending hardware changes	Overall perspective Adjudication **Enforce integrity of architecture**
Program team leaders	Progress report and review backup Gauge team capability	**Realism of decisions against** team realities
Market-segment managers	Beta site and customer reports Knowledge of competitors	**Viability of choices against user base and competitor realities**
Project document control	**Overview all documentation**	**Implement new additions in** archives
Senior system engineers	Familiarity with pending options	Fit against imminent hardware changes

List Obvious Information Gaps in Each Deliberation

At this point, the design team was asked to reflect on the analysis already done and to identify obvious gaps or cracks where information goes astray in each major deliberation. To do this, each major deliberation's topic is listed and, reviewing the information collected thus far, the team identifies evident gaps. This often proves to be one of the most useful points in the design process for gathering information. For example, the design team arranged interviews with other people in the software group. Most interviews were conducted by the consultant on behalf of the design team. Table 3 indicates how several major deliberations appeared to the team in terms of information cracks.

Table 3

Obvious Information Gaps in Selected Major Deliberations

Deliberation	*Gaps*
Stipulate system features	**Marketing information overly aggregated no tangible punch** Slow documentation transfer between hardware and software engineering Long-term strategy not clear enough to inform choices Field engineers neither receive nor contribute
PERT development and management	**Project managers do not get full information on team** Marketing not warned soon enough of delays and release impacts PERT changes poorly distributed
Declare different design editions	No summaries of recent updates Poor integration between existing standards and updates **Poor distribution of update documents to software engineers at dispersed locations** Bad proofreading of updates

Analyze Component Office Work Activities for Each Deliberation

The next phase of technical analysis had the design team scrutinize each important deliberation in terms of its component office work activities in order to highlight less obvious problems and opportunities for improvement. The design team was asked to suggest problems or improvements in a standard list of component office activities applied to every key deliberation. This exercise provided the basis for constructing an information activity matrix. The team's analysis of one key deliberation is shown in Table 4.

To summarize, socio-technical theory requires the design team to scrutinize both

the technical and the social subsystem of the workplace. Although the technical analysis is now complete, achieving the best fit between the two subsystems demands careful examination of the social network.

Table 4
Component Work Activities of a Major Deliberation:
Stipulate System Features

Component Activity	Issue
Type	WP operators find principals inaccessible for questions Erroneous documents
File	Suggested ideas not available in one spot Different drafts lost
Dispatch/receive/ sort mail	**Company mail sub takes too long to sort mail**
Dictate/read/reflect/ doodle	Reports not organized for easy scanning **Excessive wait for computer time to do financial and** time projections
Compose/draft	**Multiple authors must wait for each other's copy**
Schedule	Meetings impossible to schedule
Meet/travel	Someone always must travel for a meeting No plane to meet if unscheduled
Discuss	No good records of ad hoc technical sessions
Phone	Telephone tag between software and marketing

Step 4: Social Analysis

In the social analysis, the design team is required to examine closely the network of parties to key deliberations. These parties must effectively function as discretionary coalitions

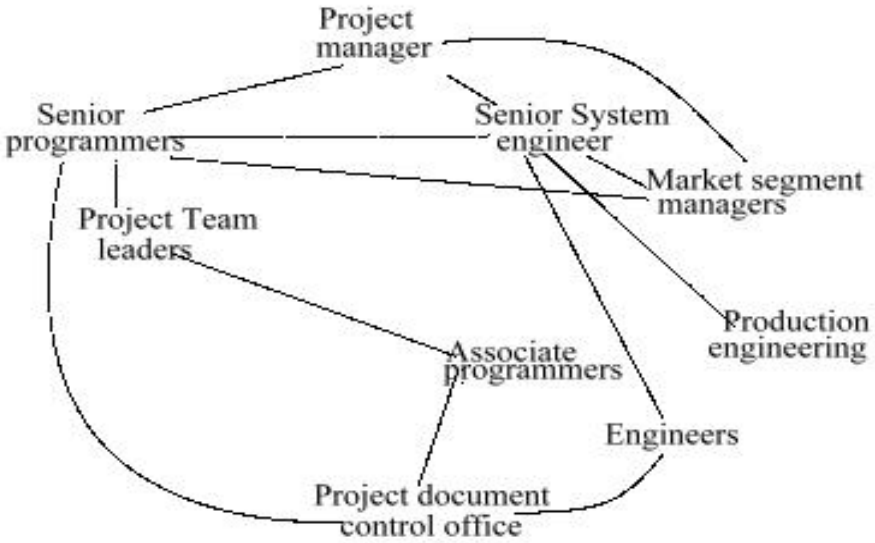
if the work system is to attain informed trade-offs and long-term success.

Depict the Role Network

To begin, the design team sketches the role network for each major deliberation. Earlier analysis helps this task proceed quickly. The list of participants in each key deliberation drawn up in the technical analysis is retrieved. Often, a design team will adopt its own diagrammatic conventions in mapping role networks. For example, different kinds of connecting lines could signify different sorts of relationships, and size of circles and distance could represent the relative power and proximity of parties. A deliberation role network drawn by the software design team appears in Figure 1.

Figure 1

Role network
for a major deliberation



Summarize Characteristic Values

Next, the design team is guided to summarize the orientations that typify each party in the major deliberations. Interviews and informal discussions can be used to confirm initial impressions. An example of one such listing is given in Table 5.

Table 5

Characteristic Values of Parties to a Major Deliberation: Declare New Design Editions

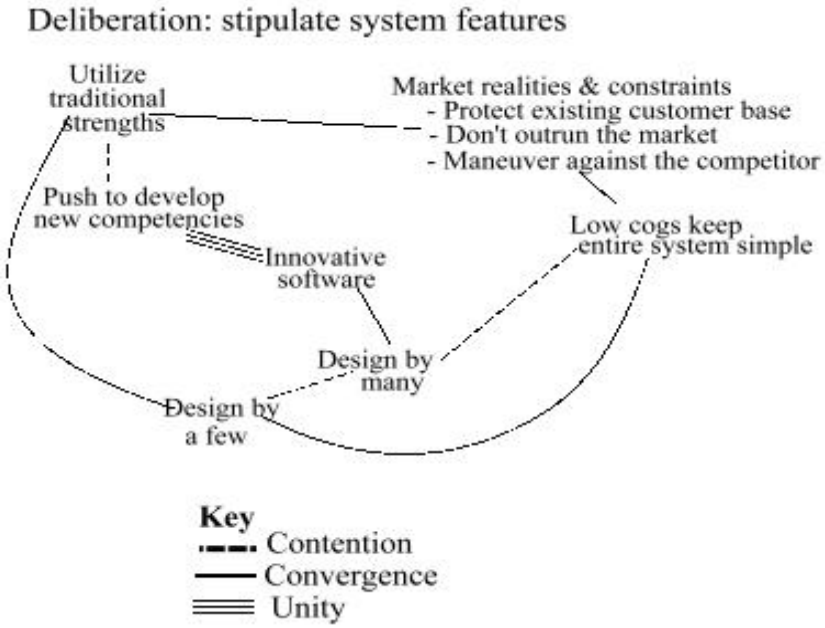
Party	*Orientation*
Project manager	Maximize hardware and software change fit Curtail features that intrude on other business lines
Senior programmers	Clearly stipulate closing dates for different editions Prevent system changes from undermining established architecture Delay edition cutoff announcements to maximize maneuvering room
Senior system engineers	Keep software changes from wagging hardware changes Get early maximum information on pending software changes
Market-segment managers	Achieve delivery date targets at all costs Make product reliably improved no great leaps Link proposed upgrades with competitor moves
Program team leaders	Do not squeeze troops with inflexible design cutoffs
Associate programmers	Get a fair shot at refining initial products Get clear cutoffs on different editions in advance Acknowledge late cuts only if they help team make specs
System engineers	Keep software changes from shaping all hardware design Resist design cutoffs that render previous solutions ineffective
Documentation control	Be sure that everyone receives and acknowledges edition cutoffs
Production engineers	Low cogs

Identify Reciprocal Values

The design team must identify divergent orientations of parties who are interdependent by virtue of engagement in the same deliberation. These orientations constitute the reciprocal values that must be balanced in discretionary coalitions. Usually, this identification is done in a separate diagram (Figure 2), but it can also take the form of value orientations added to the chart of characteristic values.

Figure 2

Reciprocal values of parties to a major deliberation



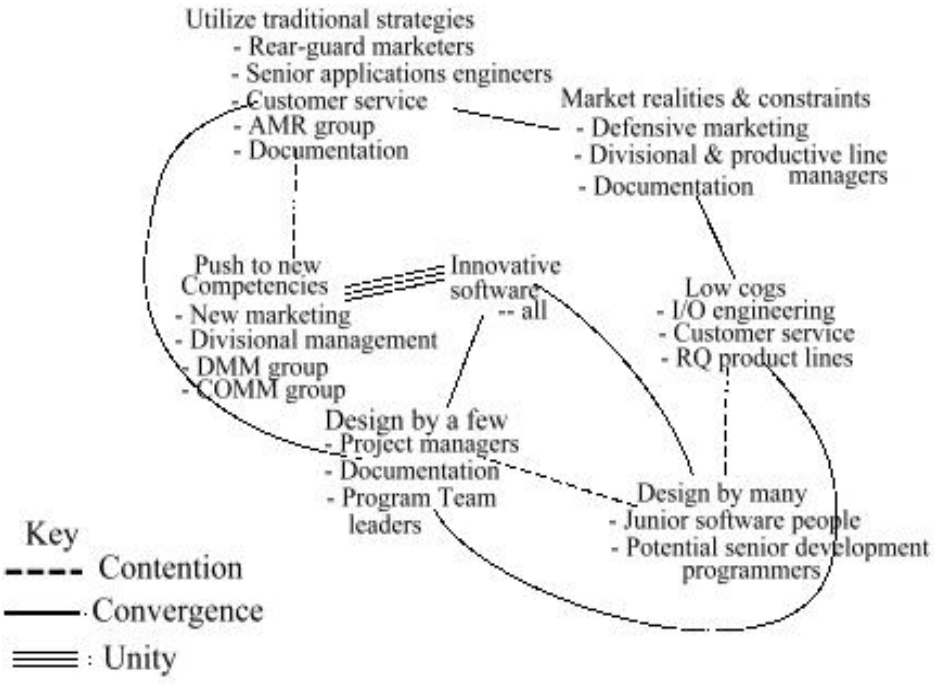
Outline Discretionary Coalitions

Finally, the design team must identify the parties who characteristically take

divergent positions. By balancing opposite interests, a discretionary coalition can guide deliberations to produce intelligent trade-offs. Figure 3 indicates how the design team set up a discretionary coalition. Through the social analysis, the design team decides what discretionary coalitions to organize in order to render key deliberations productive.

Figure 3

Discretionary coalition organized around a major deliberation



To summarize, analyzing the social subsystem of nonroutine office work reveals discretionary coalitions needed to run the deliberations identified earlier in the analysis of the technical subsystem. The method of social analysis proposed here illuminates reciprocal points

of view needed in deliberation and traces out the parties that champion these perspectives, with emphasis on a mix of viewpoints and players that can render informed trade-offs on a sustained basis.

Step 5: Work System Design

The final step of socio-technical analysis for nonroutine office work is to formulate design proposals that best match the technical and the social subsystem. The preceding analysis permits informed judgments about how the work system could be more effectively organized. In addition, socio-technical theory suggests some coherent, fundamental changes in nonroutine office work configurations that are likely to create a high commitment, high performance organization.

Charter Major Deliberations and Discretionary Coalitions

The design team can start to outline a high performance organization by chartering the major deliberations through which discretionary coalitions must strike intelligent trade-offs. Typically, this charter states the topic of a deliberation, its purpose and importance (sometimes linked with a principal organizational strategy) and the various forums through which the deliberation is pursued. At this point, the team may propose chartering deliberations not uncovered in the technical analysis. An example of a charter by the software design team is shown in Table 6.

Table 6

Charter of a Major Deliberation: Stipulate System Features

Mission	To define new product and product enhancement features that contribute to greater share in current and future markets. Vital to attain cost/functionality with timely completion. Cost is more elastic than time. New developments must permit migration from the product series.
Forums	Strategic and operation planning Quarterly marketing reviews Monthly product-line forum Customer service task force Ad hoc coffee room sessions with product marketing Weekly suggestion listing Real-time module and system manager updates
Participants and orientation	See corresponding responsibility chart, notes, and videotape.

Chart Responsibility Roles for Major Deliberations

This chart can help the design team to suggest specific forms of contribution for different coaliton members. An effective deliberation does not require that every party to it have the same form of involvement; often different forms of participation are best. A responsibility chart is an analytic tool that a design team can use to suggest how different parties can be involved in a coalition. A responsibility chart shows the preferred contribution of each party to a deliberation.

The software design team chose to do only enough responsibility charting to be

able later to teach this technique. The team thought that charting should be done by actual coalition members, perhaps aided by design team members. Practice charts were made both to train team members as facilitators and to serve as instructional material. Table 7 reproduces one such practice chart.

Table 7

Responsibility Chart for a Major Deliberation: Declare Different Design Editions

	Task					
Party	*Propose updates*	*Propose bundling updates*	*Collate revisions*	*Verify debug test*	*Document*	*Announce*
Senior programmers	A	A	I	I	—	A
Program team leaders	R	C	C	R	C	C
Market-segment managers	—	C	I	—	C	R
Project documentation control	I	R	R	C	R	I
Senior logic system engineers	I	—	—	—	—	I

R = Responsible to initiate and carry through; C = Consult; I = Inform; A = Approve; - ~ No involvement

Design Human Resource Policies that Support Effective Coalitions

The design team must suggest ways in which myriad other organizational factors

can be made to complement the work of discretionary coalitions. Aided by suggestions from their consultant, the software engineering group design team considered four major areas in which changes might be needed and made substantive recommendations for each. Among the suggested changes in each area were:

> *Compensation*. Base greater proportion of bonuses paid to senior engineers and staff above this level upon division performance. Allocate more discretionary funds to interfunctional groups for small joint development projects to promote familiarity and trust across unit boundaries. Cease to tie word processing operator pay to key strokes per hour; this was found to encourage documentation errors.
>
> *Promotion and personnel development*. Begin planning selective diagonal promotion within specialities to groom people for positions in general management.
>
> Develop programs to move all professionals out into the user environment at least twice a year. This exposure could help them appreciate marketing's problems. Provide updates of market developments to engineering professionals on a regular basis. Conversely, inform marketing on a regular basis of new capabilities that could be offered to customers. Quarterly briefings are suggested in order to provide an ad hoc form of training on viewpoints across functions.
>
> *Symbolic recognition*. Bolster the availability of documentation by creating a more direct reporting relationship between the head of software documentation and the manager of the computer system development project. Too often, limited records impede development. A more direct relation should help make

documentation a higher priority in terms of both resource allocation and compliance. Refurbish meeting rooms to make them more pleasant; especially, improve lighting and seating, and install snack machines. Current spartan meeting facilities encourage people to skip meetings. Give market-segment managers larger offices, with meeting tables and enamel writing boards. This improvement would put them on a more level status with senior engineers and allow them to convene ad hoc sessions in their offices.

Ground rules and yardsticks. Begin to inculcate ground rules for the firm's own style and method of operation. Everyone in the firm comes from a different background; thus, everybody has his or her own way of doing things. Develop programs to give people exposure to more common styles of management and to teach them the necessary skills. Initial areas of improvement include conflict resolution and summarizing information in meetings. Establish measures that underscore the importance of entire programs over the sheer value of code. Develop indices of program functionality to encourage an orientation toward performance of programs as a whole. Possible functionality measures include number of program subsections versus number of bugs between modules at late project stages; actual versus targeted performance of the entire program; and actual versus projected costs for the entire development. Developing completely accurate or conclusive indices is impossible, but ongoing efforts to measure effectivenss--the final product overall--will increase commitment to entire program systems and encourage learning.

Suggest structural changes. Specialize the phases of system design more narrowly. Close off bottom-up suggestions at a definite time and then finalize specifications based on the final choice of a few key people (programming team leaders and senior programmers, in consultation with marketing and logic design). Start a liaison group between documentation and software units to assess continually and upgrade documentation routines. This group should report directly to a group of senior programmers. Institute special councils to assess periodically the quality of each deliberation. Each council should include some top managers who would be able to sanction needed changes.

Technical enhancements. Finally, the design team will suggest changes in office technology to assist the major deliberations. First, the team should propose new information handling procedures; these specify how information is to be gathered, recorded, deciphered, circulated, reviewed and reformulated. Although the procedures stipulate what sorts of information and exchange must exist, they may not necessarily spell out how all this information handling is to proceed. At this time, the software engineering group design team recommended a number of new procedures:

- Circulation of software group PERT scheduling charts and monthly update summaries to market-segment managers.
- Regular contact between programming team leaders to chart compatibility between subroutines before entire program modules are put together; also provision of regular access to portions of

semifinished programs before release to central files.

• Quarterly updates to the entire software group on market conditions and especially on competitor activity.

Next, the design team suggests new devices to implement existing and proposed procedures. The team should construct a list of procedures that need improvement and identify both high and low technology solutions. The software group design team recommended advanced office technology for some unautomated applications, but a variety of less flashy technical changes also were proposed. Many of these would establish organized information that could later quite easily migrate to advanced communications and computing systems. A sample of the design team's technical suggestions is shown in Table 8.

Step Six: Approval and Enactment

With the design proposal finished, the team organized a special meeting to communicate findings to the steering committee. A number of changes were approved--some proposed by the design team, others suggested by the steering committee. Once a final design was settled, the steering committee and design team formulated a transition plan. The transition plan specified how the proposed design would be made operational. It recommended briefing sessions for all members of the software design group, subsequent revision and approval in the design, training in conflict resolution skills for all coalition members and periodic revision. The overall effect was to improve the quality and speed of development work in and beyond the

software group. Coordination and problem solving among professional functions improved, deadlines were met with greater frequency and the work setting became less stressful.

Table 8

Selected Technical Recommendations

Suggested enhancements	*Gap or procedure*	
	Program Evaluation and Review Technique (PERT) chart tracking	*Regular Program Team Leader (PTL) contact*
Low technology enhancements	**Local copying (protocol for** common update notation)	Bigger boards and tables in PTL offices
	Make review part of program team leader sessions: weekly and midcourse project reviews	**Polaroids to capture board** info.
High technology enhancements	Interface costing estimation for software and PERT programs	Computer messaging to end telephone tag
	Weekly updates via electronic mail distribution	Provide small portable terminals for remote taps
	Digital store and forward voice messaging	**Phone conference between** Bendrel and Wallace sites (with slow scan TV?)