Received November 1996
Revised December 1996

Supporting the social processes of software development

Steve Sawyer
Syracuse University, New York, USA, and
Joel Farber and Robert Spillers
IBM Corporation, Santa Theresa Laboratories, California

This paper addresses one instance of software developers attending to their own work-related needs. Behind this paper lies our belief that one key to improving software development is to make it easier for the developers to work together. This perspective is shared by many (e.g. Boehm, 1981, 1987; DeMarco and Lister, 1988; DeMarco, 1995) and our purpose in this paper is to describe one site's approach to developing a facility that allows team members to work together. Specifically, we report on the use of a computer-supported meeting facility that we call the "team room". In presenting the team room and its use we address four questions:

(1) What is the team room?

(2) How is the team room used?

(3) What makes it useful? and

(4) How is team room use affecting software development performance at this site?

Computer-supported meeting facilities are common (see e.g. McGrath and Hollingshead, 1994) and are being used to support software development (Carmel, George and Nunamaker, 1992; Tyran, George and Nunamaker, 1993; Liou and Chen, 1993-94). However, the development and use of the team room at this site is different from these previously reported experiences for two reasons. The first reason is that the team room we describe in this paper is a "home-grown" system. That is, the facility grew out of the software developer's own efforts to help themselves to work together. The second reason relates to how the team room is used to support software development. Compared with existing electronically supported meeting rooms (EMS), the team room imposes little structure on how the group works (Mcleod, 1992) and does not require facilitator support (see Growohoski *et al.*, 1988). In this way it does not serve the

This work has been supported by IBM contract STL92135. The generous assistance of Harry Campbell, Mike Pauser, Mike Dockter, Andrew Peterson, Kevin Seppi and Dave Tolleson have made this paper possible.

more stylized role of process support for which many EMS are designed (DeSanctis and Gallupe, 1987).

Software development as production

Much of the present focus on improving software development seeks to make it more of an engineering discipline by concentrating on improving the production process (Humphrey, 1988 and 1995). Production aspects of software development include methodologies, techniques and tools. Examples of these include integrated tool suites in improved development environments (using integrated CASE tools) and more powerful software languages (such as C++). This thrust draws on more powerful techniques (like JAD), advanced process control methods (such as SEI's Capability Maturity Model or the IS0 9000 series of standards) and more managerial skills (such as risk and project management).

Despite these efforts, the software development process must improve (Davis, 1996). Viewing software development as more than a production process opens new ways to achieve these improvements. One view is to routinize development as a series of scenarios and then to automate them. This leads to a flexible, but standardized, software development process which is exemplified by "workflow" models (Brown, 1995; Chroust and Bergsman, 1994). A second thrust focuses on how the work is being done. This focus attempts explicitly to account for the social and behavioral aspects of software development in the production methods (Curtis, 1989). For example, a better understanding of how development is enacted can come from an account of worker behavior (Sachs, 1995; Suchman, 1983 and 1995). In this paper we opt for this social-behavioral approach. Our goal is to describe how software developers have used their own work practices to guide the evolution of a facility that allows them to work together more easily.

The paper continues in four parts. The first part lays out the motivating forces behind the team room's evolution at this site. The second part describes how we conducted our study. The third part summarizes the findings from our analysis of how the team room supports the process of developing software. The fourth part of the paper discusses some implications, both anticipated and unexpected, of team room use.

Issues with developing software

To understand the context underlying the team room's emergence, this part describes the issues facing the developers at one software development laboratory. The site is one of several software development laboratories and part of a multinational hardware and software development corporation. The site's products are mostly large systems software that can operate on multiple platforms. Many products are industry standards and some have been in the market for nearly three decades. More than 1,800 developers work at the site. Nearly 89 per cent of the developers have college degrees, with 37 per cent of these also having masters or doctorates. The employees average more than 15 years in software development, more than 11 years with the company and more than seven years at this site.

Like other large systems developers, the site faces increasing competition. Product time-to-market issues and quality are often competing pressures. The market life of each software product being developed is also shortening. Even so, long-term maintenance demands increase with each new release. So, by the early 1990s the site's product complexity grew quickly, while development became more cumbersome. This combination led to difficulties in maintaining existing products and slowed the pace of new product development. Defect rates for some products rose above targeted levels and customer satisfaction with some product lines declined. Concurrently, developers were increasingly unhappy with the way their work was structured and with the pressures they faced at work. They were working harder, spending more time at work and seeing fewer results. Senior managers at the site also were under tremendous pressure to create new products and extend revenue streams on existing lines.

These intertwined influences – market pressure, product pressure, work life pressure and managerial pressure (including changes in senior management) – led the site's leadership to rethink its approach to developing software. In 1992 the site began a two-pronged approach to revamping software development. The first prong was a focus on team-based development and a move away from the functional/project matrix management structure. In doing this, the second prong was to leverage available software tools and existing development techniques to facilitate their work.

The team room arises

Facing the need to get software development specialists with strong and divergent views to interact on complex problems, one team began dragging a PC and an overhead projector into a conference room. At first this arrangement was crude and temporary – a laptop or workstation wheeled in on a cart and an LCD image projected onto a pull-down screen. While seemingly simple, this meant that, instead of staring over the shoulder of one developer to look at her screen in a small office, team members could see more easily the LCD image projected on the wall. Work products (e.g. software modules, status reports and documentation) were carried into the room on floppies and loaded onto the machine as needed (the computer in the room was not networked). However, LCD-screen resolution was poor and the lack of network access complicated things. The room was also uncomfortable since the computer warmed the room quickly and the reflected glare from the lights made the screen image even hazier.

Despite these limitations, this crude screen-sharing made it easier for people to work together on the same product, at the same time. Other teams began experimenting by adding various tools and trying out other configurations. Soon, the computer in the meeting room was connected to the local area network so that files and software tools were more accessible. Bigger screens with higher resolution were installed. One-by-one, rooms were remodeled to accommodate a computer workstation and provide more comfortable working conditions. For instance, improved air-conditioning systems, more ergonomic seating, indirect lighting and sound-dampening material were installed. This made it easier to work for several hours in a darkened room full of people with a computer running.

Other teams took available technologies and adapted them for use in the room. For instance, electronic white boards were installed. Facsimile machines were added – and images (from the white board) could be transferred (by fax) to the computer. Additional phone lines were added for conference calls. A video camera was added so that the screen could be videotaped. Tapes provided a record of the work and a record of the discussions and decision making. These tapes also were expected to serve as training aids, to record decision processes and to act as meeting minutes (so those not attending could "catch up").

Developers began pushing management to help improve these rooms and to get more of them. Since the rooms represented small investments (less than $25,000), this was possible. The site's in-house R&D group worked with the development teams by getting better electronics (like better screens and specialized lighting) and improving the room's ergonomic design (with better chairs and non-reflective surfaces). The R&D group worked also with developers to identify and adapt several existing commercial software products for use in the facility.

The team room

The present team room is a conference room redesigned to support intense and/or extended team-based work sessions (see Figure 1). The single

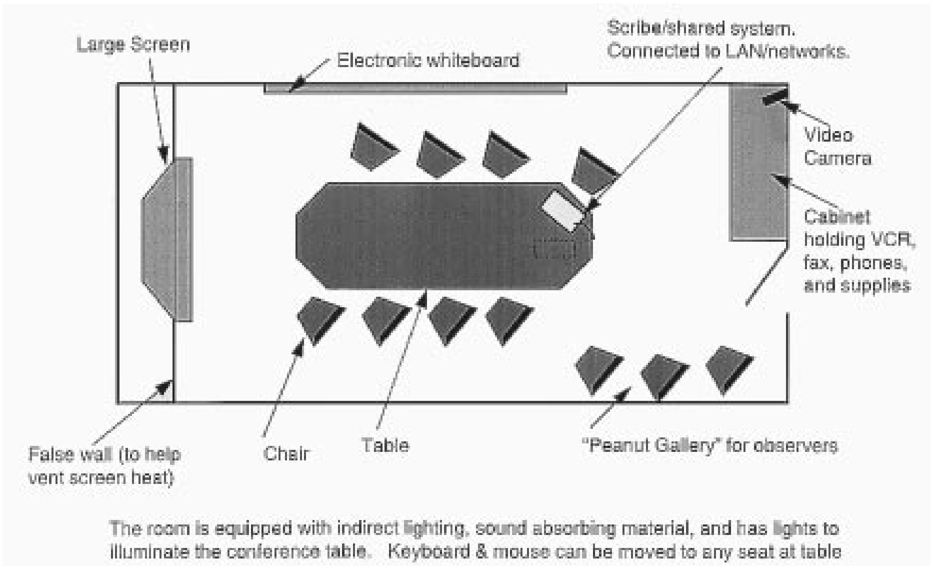
Large Screen
Electronic whiteboard
Scribe/shared system. Connected to LAN/networks.
Video Camera
Cabinet holding VCR, fax, phones, and supplies
False wall (to help vent screen heat)
Chair
Table
"Peanut Gallery" for observers
The room is equipped with indirect lighting, sound absorbing material, and has lights to illuminate the conference table. Keyboard & mouse can be moved to any seat at table

Figure 1. The team room: an overhead view

workstation in the team room is connected to the organization's local and wide-area networks. This connection provides access to software tools and the team's work products during a team meeting. The room's dominant feature is a large screen at one end which is readable from any point in the room. While there is only one keyboard, the screen image serves as a shared work space. This makes the single workstation a shared system.

Software products accessible from the room's computer include the tool suite which is available at an individual's workstation. However, the team room environment has led to changes in the suite's use. For example in the team room commercially available tools designed for group work, but not used much at individual workstations, are used extensively. A group editing tool allows layered comments and viewing of comments – and for comments on these comments (resembling what is reported in Olson *et al.*, 1993). Another tool allows two workstations (on the same network) to see the same screen and to work jointly with what is on that screen. The R&D group also modified an existing software tool for scheduling and tracking access to the team room. This modified tool also allows the team room's "owners" (the department that paid for the renovation of the existing conference room) to set up open access times around their schedule.

The first dedicated team room was available in early 1992. The site now has 22. Each has one computer workstation connected to the organization's network(s) and a large common screen. These team rooms also have supporting audiovisual equipment such as video cameras, VCRs, facsimile machines, electronic white boards and phones. The facility is ergonomically designed to have sound-dampened walls, low reflective surfaces and special lighting (e.g. lights that illuminate the table without creating screen glare).

The common display uses a high-quality projector. Since these projectors produce heat and noise, typically a false wall is built to support the large screen and the projector sits behind the false wall. The large table in each room is constructed to allow the keyboard, mouse and screen to pass easily between seats. However, the workstation typically sits at the end of the table away from the common screen. The keyboard is shared among the room's occupants in one of two ways: rotating or dedicated scribing. With a rotating scribe, people take turn typing as their expertise or skills are needed. With a dedicated scribe, one person responds to requests and direction from the group.

One key reason for the growth in the number of team rooms is the perception among developers that the team room makes it easier for people to work together. So, developers from a team using the team room would tell their peers in other teams. Lured by the promises in these stories, team after team clamored to use team rooms. This peer-to-peer propagation spread through the development teams at the site. A developer from the R&D group called this "… biologic; an infection. A guy from one team sees how useful it can be and infects his team with the idea. Pretty soon they've caught the bug, too".

Studying the team room's use

This part describes the research methods used to study the team room's use. Developers and managers wanted to assess how team room use enabled software production. They sought direct observation of their work as it occurred, quantifiable data on use and performance, and a synthesis of the anecdotes and stories of team room users. To that end, beginning in 1993, we spent 18 months studying the site's software developers doing their work in their native environment using (and not using) the team room. Over these 18 months we collected data on team room use and software design team performance using a combination of methods.

With the support of the team members and their managers, we used semi-structured interviews to gather data from the members and the managers of 45 software development teams. We formally interviewed 56 people. Many of these people spoke with us again, often several times, over our observation period. Interviewees included developers and managers, including senior managers. This group included people who had never used the team room but required outputs of team room-supported teams and many of the team room's early proponents and chief advocates. These people reflected a range of team room use – from little or none to extensive. We met also informally, or in organized response sessions, with another 153 developers and managers from the 45 teams following our survey data collection.

Team size ranged from four to 14 people, with no detectable relationship between team size and room use. The teams in this sample reflect the range of team room usage. Nearly one-third of the teams used it every day, with half the sample using it at least weekly or more. The other half of the teams used the team room no more than several times monthly, and two teams never used it during the 18 months we collected data.

Survey-based data were gathered from 40 teams (and 128 respondents) at the site. Five teams were not surveyed: three declined, and two were performing software support rather than development. The surveys were developed from existing scales and pilot-tested at the site (Dillman, 1978). We used these to gather demographic data about the teams and team members, the features and functions of the team room that team members used, and self-reported team-level performance. Questions were posed at the team level as this is the level of analysis. This is also the level of measurement and the level of theory (Klein *et al.,* 1994). The self-reported team room usage data was collected so as to allow it to be compared with data collected from the organization's archives on team room use.

We collected data also about performance from stakeholders for each of these teams. This was done using a structured survey in a phone-based interview using the scale developed and used by Henderson and Lee (1992) in their study of software developers. Stakeholders for these teams are people who, while not on the team, depend on the team's work (Lee *et al.,* 1991; Seidler, 1974). These might be user managers, senior development managers, and/or senior customers.

For the interview and observational data, the transcripts and field notes were content-analyzed to highlight themes and recurrent issues. We used these as the basis for follow-on interviews. In the debriefing (offered to all 45 teams and accepted by 28) we asked additional questions to reflect on the findings from both the qualitative and quantitative data analysis.

Analysis

In this part the data analysis is presented. This analysis is done to address three questions:

(1) How is the team room used?

(2) What makes it useful? and

(3) How is team room use affecting performance?

The developer survey provided data on demographics, team-member use of the team room, and self-rated performance measures of the team's effort. Archival records provided data also on team room use over time and on product-defect rates (a measure of performance). Additional performance data are based on surveys of stakeholders. Interviews provided data on team room use and the role it plays in developing software. Individual data are aggregated to the team level for analysis (James, 1982).

To address how the team room is used, archival data on room use is plotted over time. This is compared with the self-reports of team members responding to questions on how long they have been using the room and how often they use the facility. The multiple team member responses on duration of use agree in 90 per cent of cases with archival data. They agree 93 per cent of the time on frequency.

To address the last two questions we draw on both survey and interview data. The developer survey provides data on how, and how often, team room functions are used. Data from the interviews are used to form an explanatory effects matrix (Miles and Huberman, 1994; Miller, 1982). In constructing this matrix to address what makes the team room so useful, the various reasons for team room use, drawn from the interviews, form one axis. The various outcomes or effects of team room use, also drawn from the interviews, form the other axis. The matrix is completed by developing links between the two based on the interviews. For example, from the field notes and interviews, we looked for how the team rooms are used. As we organized these, categories or types of use emerged. These types of use formed one axis and supporting statements formed the other.

A matrix for how team room use affects performance is also developed. For this matrix one axis is for reasons for usage and the other axis is for performance effects ascribed to usage. Additional performance data are drawn from archives (the aggregate product-defect rates) and from stakeholder responses (regarding both product and team performance). The following sections summarize the findings.

How is the team room used?
Our analysis reveals three findings regarding team room use. The first finding is the extensive use of the team rooms. The second finding is that this use varies widely as to both how much, and in what way, the team rooms are used. The third finding is that using team rooms has become an accepted part of developing software at the site.

Usage records show extensive team room use. The number of meetings, the booking rate, and the utilization rate of the team rooms are presented in Figure 2. During the 18 months of observation the number of meetings held stayed steady around 50 per month per team room. However, over this time, the number of team rooms grew from 14 to 22. So, the total number of team room-supported meetings grew from 14 rooms used about 50 times per month (700) to 22 rooms used about 52 times per month (1,144). Over this period the average booking rate (the times when rooms are sought) averaged 130 per cent of aggregate room capacity (defined as the period from 8 a.m. to 5 p.m. on workdays). Peak periods, such as late winter, see booking rates at 160 per cent of aggregate room capacity. This overbooking means requests for use exceed aggregate room capacity.

Team room use, the degree to which the rooms are used, shows a trend over the 18 months of about 75 per cent. Less-than-full use is due to both room downtime (for maintenance or upgrades) and unusable blocks of time (e.g. only 30 minutes free between scheduled meetings). Thus, data on use is conservatively measured since it does not reflect the unscheduled use of these free times, a common occurrence.

We saw teams adjusting their work schedules to use the facility after normal business hours: arranging to come in early or stay late to use a team room. For instance, one project manager said his team would not start new development work until they could get a team room. He felt the delay caused by waiting for access was always recovered in the work done during the meeting. Said a senior

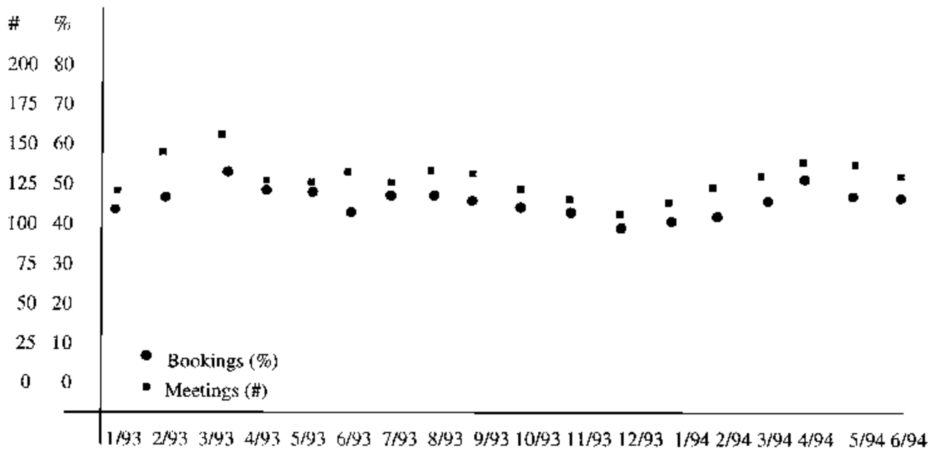

Figure 2.
Utilization and booking rates for team room

manager: “Everyone has used the [team room] at some point”. While this is not exactly true, it represents the common perception of the value placed on the team room.

Types of team room-enabled meeting vary greatly. Since the facility is a collection of simple technologies designed to support team work, use varies by team and by task. The team room is used in the range of meetings typically associated with software development, such as requirements and specification definition, code writing and debugging, code review, project status and project postmortems. For each type of meeting, team room use differs. Table I outlines the way the facility is used and what software tools/features are most often used. Respondents were asked which software tools/ features they used from a set that included:

- the group editor;
- electronic white board;
- site-link software;
- presentation graphics;
- word processing;
- coding/debugging tools;
- project management tools;
- fax/voice connections;
- video-camera; and
- other (an open-ended response).

For example, in requirements and specification meetings, teams use the whiteboards and computer-based drawing/presentation tools more than any other features. These meetings are characterized by constant debate, high levels of conflict and much interaction. For coding and debugging meetings, the code is displayed on the main screen for all to work on together. This code can be run on the screen and debugged in real time with the entire team (or those

Table I.
Room usage and feature usage

Used for	Use as %[a]	Features used (as %)[a]	
Scheduled meetings	81	Group editor (67)	White board (90)
Ad hoc meetings	63	Group editor (40)	White board (95)
Brainstorming	57	Group editor (15)	White board (90)
Teamwork	24	Group editor (67)	White board (90)
Remote site links	9	Group editor (10)	Site link (100)
Privacy/solitude	6	n.a.	
Other	18	Project (35) Management	Presentation (15) Graphics

[a]Can check multiple uses, only top two reported

who needed to be involved) sitting together. Instead of discussion in the meeting and action between meetings, using the team room allowed developers to debug together during the meeting. Said a team leader: “Now the experts talk and we can listen. Used to be everyone had an opinion. Most of us talk less, lots more gets done.” A senior technical lead said: “Using the [team room] means I can lead people to the solution I know will work. They can argue, I can show. And we get where I want to go”.

Code review meetings typically use the group editing tool. This allows multiple comments to be made on the same document. Typically, comments are made to the electronic copy by people working from their individual offices. These are brought up and discussed in a meeting. Using the team room’s shared screen allows this discussion to happen in the meeting. Instead of huge paper dumps which demand extensive premeeting work, the team room allows people to come in and run through the document during the meeting, making changes and discussing points along the way. Said a senior developer: “We review it there. It’s done when we leave, not like a meeting – it’s work getting done”.

The team room also enables effective use of project management software, agenda sharing, access to work products during meetings and meeting notes’ distribution (which can be done electronically at a meeting’s end). One product manager, faced with getting senior technical people from the performance, quality and access departments together to work on fixing a sluggish and buggy section of code, decided to use the team room to get these experts together. She sent out an electronic copy of the agenda and moderated a small “flame war.” Following this the experts came, she said: “prepared to talk about issues, and demonstrate their points”. Several of the topics were cleared before the meeting. In the meeting she had the agenda and code on the screen. They ran code as they talked through issues, coming to a resolution in less than 24 hours from the initial e-mail.

Another software tool allows team rooms in geographically separate locations to share the same work product on the screen. Thus, people in two team rooms in the same building, or on separate continents, can work together simultaneously on the same product. Using the phone as the audio link means this work can be both visual and audio. For example, groups of lawyers from two different sites used the group editor and common screen link software tool to work together on a software licensing agreement without having to make the transatlantic flight. This is different from video conferencing, since the two groups worked on the same document(s) in real time; and separating phone lines from the computer link provided a useful redundant connection when the screen at one site froze.

Using the team room is an accepted part of working at the site. The facility is now integrated with the site’s software development process. The team room has become a part of the fabric of development – just like code reviews and project status meetings. For example, facility maintenance, usage tracking and scheduling tool support are now regular internal information systems support group functions. In our follow-up interviews, developers wondered why we were so interested in the team room: “Oh that,” said a developer, “it’s fine, we use it … Why not? It helps.” That team room use is no longer topical supports it’s acceptance, and embeddedness, in the site’s software development process. Said the quality guru: “We [managers] don’t have to push, they [the teams] ask to use it [the team room]”.

What makes the team room so useful?

The team room facility is based on simple technologies. There is no special software; it demands no special training and does not require external facilitation. Also, there is no mandate to use the team room. However, it is used widely. Our interviews and observations suggest that the team room is used because it provides a shared work space. And, this work space adapts to how each team uses it. The team room becomes a team space. This team space concept is highlighted in at least two ways. First, interviews and observations suggest that the screen serves as a focal point for all people in the room. Second, these interviews and observations show that work is done in the room during the meeting.

When people use the team room, they focus on the screen and on the work product on the screen. Since the work in the team room is projected on a shared screen, it “belongs” to the team, and individual ownership is diffused. In this way the shared screen acts as a social buffer. Two developers may be talking to each other, but they do so staring at the screen. This means that when disagreements arise there is no need to directly face the person who did the work. Thus, in the team room it is easier to disagree with the work on the screen and make changes to it without having to negotiate directly with the person who wrote the code or document. For example, in observing team room use we often saw people venting their anger toward the work on the screen and not toward each other.

The team room enables social interaction by providing a buffer between people. In team room meetings people face the screen. Instead of face-to-face, work in the team room becomes face-to-screen-to-face. While conversation flows, people rarely face each other. These team room meetings look much like watching a movie, save that the audience is actively engaged in making the movie as they watch – a truly interactive show. In our interviews, this appears in subtle ways. People speak of being more comfortable to talk about problems with the product on the shared screen. Says a developer: “...people can say what they need to say. Stuff gets done there. Not just talking”.

Work gets done in the team room. An original intention of the team room had been to reduce meeting time – a production-focused efficiency issue. Instead, developers report that per-project meeting time has remained constant. People find they can access work material and get work done in team room meetings. This is contrast with most typical meetings, where the majority of work is done between meetings. In the team room developers can try out ideas by running the code while working in concert as a team. Often, disagreements on how to make changes in code get resolved by running the variants for all to watch.

Large paper dumps (common in code review and documentation review meetings) are no longer needed; the document is brought up on the screen and editing is done as a group. “I don’t hate the team room like I hate meetings,” said one junior developer.

How is the team room affecting software development performance?
At this site, two outcome measures important to the software developers are product quality (the number of defects reported) and developer work satisfaction. These are also the major criteria for assessing improvements in software development performance. From 1992 through 1994, the number of defects in the software products at this site fell by 50 per cent. This is not directly attributable to the team room, for two reasons. The first reason is that team-building and process-quality initiatives were coexistent efforts. The second reason is that we cannot relate reported defects to specific project teams since many teams work together to build integrated products. However, the team room provides the vehicle for these efforts; serving as the workplace for the developers whose products posted such large quality improvements.

Analysis of the survey-based data shows relationships between team room use and higher levels of developer satisfaction and improved product quality. For teams which used the team room extensively (more than once-weekly), developers reported above-average improvements in work satisfaction and product quality compared with those teams which used the team room less. From the survey data, the mean for the 40 teams is 5.23 on a seven-point scale, with “7” scaled as most satisfied. The mean for the 20 teams using the team room most is 6.01. Stakeholders report higher levels of product quality for teams using the team room most. The zero-order correlation between stakeholder-rated product quality and greater-than weekly use of the team room is 0.4962 with a p of 0.031 for the 20 high-use teams. Since these stakeholders are not part of the teams, they are unaware of using (or even the existence of) team rooms. We interpret this to mean that, while the stakeholders are unaware of the teams’ use of team rooms, they notice an improvement in product quality for those teams who use team rooms extensively.

The team room in software development

In this fourth part we address some emerging issues arising from the evolving use of team rooms. To frame this part of the discussion, we repeat that a central aspect of a team room’s value is that its use is so varied. For example, some teams spend the greater part of most days in these rooms, while other teams also meet in them regularly, but less frequently. Independent of frequency of use is the question of how these rooms are used. For example when some teams use a team room people flow in and out of these ongoing meetings, while other teams have fixed attendance at the meetings. Some teams have structured agendas when using team rooms, while others work in a more open format. For some teams, using team rooms leads to more people communicating, while other teams see a reduced amount of discussion. Several teams rarely, if ever, use a team room. The interviews and observations suggest that teams using the team rooms spend more time on task during meetings than do the teams which rarely, or never, use the team rooms.

A team room seems to provide a mechanism for group co-ordination and communication. The software tools and other technologies are mundane. It is their collective use that serves as the enabler of these software development teams. These teams, composed of functional specialists working together, can work together more easily and more productively. We believe the success of the team room lies in its ability to support the social interactions of developers. This allows the developers to focus more easily, and more successfully, on producing software.

The most common signs of the positive effects are higher levels of developer satisfaction and improved product quality. McGrath (1990) and McGrath and Hollingshead (1994) argue that a team operates at three levels: work production; social maintenance; and ego support. While these are distinct levels for analysis, they are highly interrelated in practice. Our findings suggest that using team rooms supports the social maintenance of teams and that this is, in turn, linked with the other two levels. The rooms' use puts team members together, which encourages informal conversation. The rooms' layout makes it easier to deal with social conflict, further encouraging interaction. This is abetted by the simplicity (and low cost) of the team rooms' construction. The apparent direct effect on both developer satisfaction and improved product quality by using team rooms may actually be the result of two aspects of their use: first, as a way to enable the social/behavioral processes of working together; and, second, as a way to improve the productive efforts of the team. These are the unexpected effects of use, and new forms of working together deserve additional attention.

Unexpected effects of team room use. One unexpected outcome has been the increased reliance on the team rooms to serve as a mechanism to deal with intra-group conflict. Intra-group conflict is endemic to work groups (see, Pondy, 1967; Thomas, 1975) and to software development groups in particular (Robey, 1984; Walz, Elam and Curtis, 1993). Intra-group conflict is neither bad nor good; its presence means that disagreements about both the nature and the purpose of information and decisions arise between team members as they interact. Since intra-group conflict is present in software development teams, having the team rooms makes it easier for the developers to deal with conflict. However, this is not done by improving their conflict management skills or providing them with feedback on their group process (Losada *et al.*, 1990). Instead, the screen serves as a buffer between people. Team members come to rely on indirect interpersonal discussion. This may be what the software developers like about using the team rooms. "We still have all that people stuff, it just goes easier in [the team rooms]," said one junior developer. In this way, we see the team room serving as a vehicle towards achieving what has been called "collective mind" (Weick, 1995; Weick and Roberts, 1993). Collective mind focuses attention on the "heedful" performance of contribution, representation and subordination (Weick, 1995). The attention is focused on the group, not the person.

Another unexpected outcome has been the role of the videotape meeting archives. At first these tapes were touted as a way to record the meeting process. Tapes allowed for the capture of decision making. They could also serve as a form of tutorial (where junior developers could see how senior developers deal with specific issues) and as a record of work or a team memory (so absent team members could "catch up" quickly). However, the tapes became a symbol of corporate memory. That is, with no formal policy as to access and no idea of how they could be used for rewards (or punishments), team room users typically opted not to use the video camera and audio tape features available. The existing tape archives became well-guarded repositories. Access is limited to a very few and this access is only granted after all taped members agree. Finally, the tapes are no longer stored centrally, but are kept by the managers of the teams which were taped.

New forms of working together are appearing. The most explicit example of this is the increasing use of the team room to support software development. A second example is the appropriation of the team room for other types of work. This includes uses like the team of lawyers from several sites collaborating on a contract. Another appropriation is development managers using the team room as a planning facility. An agenda was e-mailed to all meeting participants. The agenda, and the comments, were then brought up at the meeting. All participants knew what to expect and were prepared actively to discuss and investigate each issue.

A third and more radical example is provided by the R&D group at the site. They reconfigured their work space to reshape their entire environment to be more "team room-like." Known as the "team room 2", this space replaces 14 small offices and a small conference room with a large open area. The major difference from the flexible office so common these days is the proliferation of large common screens around the room. The LAN supporting this environment allows any developer to toggle his or her personal machine's screen to a common screen for sharing. The work space is busy, visually stimulating, and still adapting to the users of team room 2. For instance, small spaces are placed at the edges of the common area. These are available for private calls and "quiet time." The manager also maintains a private office for confidentiality and meetings. Finally, a "normal" team room is part of the space.

Software development as discussion

While the software suite of the team room is identical to the software suite at any individual's workstation – including access to work products – use is different. By sharing the computer, the same suite is used differently. And, save for the scheduling software, team rooms have no unique software. The facility provides a venue for teamworking, designed by developers to enhance support of their software development needs.

Malone, Lai and Fry (1995) have argued that radically tailorable tools for co-operative working allow "users to create a wide range of different applications by progessively modifying a working system", (p. 178). Their context was asynchronous meetings and specific tool development. Broadening this perspective to see a system as including the social structures puts the team room in perspective as a radically tailorable tool. In this way, the team room becomes a conduit, serving as a vehicle for the teams to work. The direct effect is to make it easier for developers to work together; enabling the production aspects. So, software development improvements at this site have emerged without increased engineering. Rather, they have emerged due to increased discussion.

This can be seen as adaptive from a structurational perspective (DeSanctis and Poole, 1994). Structuration theory asserts that social and physical structures influence each other over time (Giddens, 1984; Giddens and Turner, 1987; Orlikowski, 1992; 1993; Orlikowski and Robey, 1991). Social structures can be norms, values, roles and ways of interacting. Physical structures include the use of space, delineation of boundaries and the role of technologies/machines. This interactive structuring is done by selective (and often unconscious) decisions on what and how to use these structures. That is, since the social structures are formed by, and themselves form, the physical structures of use, a facility that provides a way to adapt easily to various uses would be popular. Unlike highly structured computer-supported meeting rooms, the team room imposes minimal constraints on the participants (e.g. Growohoski, *et al.*, 1988). McLeod and Liker (1992) also have argued that low-structured computer-supported meeting rooms are more adaptable to the variations in how teams choose to work.

This study suggests that when the developers at this site were faced with production problems, they focused their efforts on making it easier to work together. What arises from this reflective effort for self-improvement is a simple facility, gathering a set of common technologies in a way that easily adapts to the way the software teams work at this site. The team room is not a production platform. It is a place to discuss. The team room's design makes it easier to work together. The team room provides a vehicle to support the social and behavioral aspects of software development, and this is reflected in the improvements to software production at this site. The homemade facility, the infectious mode of use, and the anticipated and unexpected outcomes of use, reflect the desire of the developers to do better work. This is the essence of the story. Will the team room have the same effect at other sites? Currently, it is too early to tell.

Team rooms now exist at several other sites. Our involvement with these sites is limited and our evidence anecdotal. However, the use, and reactions to use, at these sites seems to follow the team room's trajectory at the original site. Finally, in this work, we resisted the desire to quantify the contributions of each element of the team room. Like eating water with a fork, understanding the contribution of each piece of the team room will be difficult and may have little reward. What we can say is that at this site the team room supports software development. The facility supports the interaction of developers and makes it easier for them to work together in producing software.

References

Boehm, B. (1981), *Software Engineering Economics*, Prentice-Hall, New York, NY.

Boehm, B. (1987), "Improving software productivity", *Computer*,Vol. 2 No. 1, pp. 43-57.

Brown, S. (1995), "The fall of software's aristocracy: realizing the potential of development", in Leebaert, D. (Ed.), *The Future of Software,* MIT Press, Cambridge, MA, pp. 157-75.

Carmel, E., George, J. and Nunamaker, J. (1992), "Supporting joint application design with electronic meeting systems: a field study", *Proceedings of the 13th International Conference on Information Systems*, pp. 223-32.

Chroust, G. and Bergsman, J. (1994), "Workflow systems", *Proceedings of CON '94, Workflow Management: Challenges, Paradigms and Products*, pp. 291-3.

Curtis, W. (1989), "Three problems overcome with behavioral models of the software development process", *Proceedings of the 11th International Conference on Software Engineering*, pp. 398-9.

Davis, A. (1996), "It feels like *deja vu* all over again", *IEEE Software*, Vol. 16 No. 4, p. 4.

DeMarco, T. (1995), *Why Does Software Cost So Much? And Other Puzzles of The Information Age*, Dorsett House, New York, NY.

DeMarco, T. and Lister, T. (1988), *Peopleware: Productive Teams and Projects*, Dorsett House, New York, NY.

DeSanctis, G. and Gallupe, B. (1987), "A foundation for the study of group decision support systems", *Management Science*, Vol. 33 No. 5, pp. 589-609.

DeSanctis, G. and Poole, M. (1994), "Capturing the complexity in advanced technology use: adaptive structuration theory", *Organization Science*, Vol. 5 No. 2, pp. 121-47.

Dillman, D. (1978), *Mail and Telephone Surveys: the Total Design Method*, John Wiley & Sons, New York, NY.

Giddens, A. (1984), *The Constitution of Society: Outline of the Theory of Structur*e, University of California Press, Berkeley, CA.

Giddens, A. and Turner, J. (1987), *Social Theory Today,* Stanford University Press, Stanford, CA.

Growohoski, R., McGoff, C., Vogel, D., Martz, B., and Nunamaker, J. (1988), "Implementing electronic meeting systems at IBM", *MIS Quarterly*, Vol. 20 No. 4, pp. 420-30.

Henderson, J. and Lee, S. (1992), "Managing I/S design teams: a control theories perspective", *Management Science*, Vol. 31 No. 8, pp. 757-77.

Humphrey, W. (1988), *Managing the Software Process*, Addison-Wesley, Reading MA.

Humphrey, W. (1995), *A Discipline for Software Engineering*, Addison-Wesley, Reading MA.

James, L. (1982), "Aggregation bias in estimates of perceptual agreement", *Journal of Applied Psychology*, Vol. 67 No. 2, pp. 219-29.

Klein, K., Dansereau, F. and Hall, R. (1994), "Levels issues in theory development, data collection, and analysis", *Academy of Management Review*, Vol. 19 No. 2, pp. 195-229.

Lee, S., Goldstein, D. and Guinan, P. (1991), "Informant bias in I/S design team research", in Nissen, E., Klein, H. and Hirschheim, R. (Eds), *Information Systems Research: Contemporary Approaches and Emergent Traditions*, North-Holland, Amsterdam, The Netherlands.

Liou, Y. and Chen, M. (1993-1994), "Using group support systems and joint application development for requirements specification", *Journal of Management Information Systems*, Vol. 10 No. 3, pp. 25-41.

Losada, M., Sanchez, P. and Noble, E. (1990), “Collaborative technology and group process feedback: their impact on interactive sequences in meetings”, *CSCW ’90 Proceedings*, pp. 53-64.

Malone, T., Lai, K. and Fry, C. (1995), “Experiments with OVAL: a radically tailorable tool for cooperative work”, *ACM Transactions on Office Information Systems*, Vol. 13 No. 2, pp. 177-205.

McGrath, J. (1990), “Time matters in groups”, in Galeghar, J., Kraut, R. and Ergido, C. (Eds), *Intellectual Teamwork, Social and Technological Foundations of Cooperative Work*, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 1-23.

McGrath, J. and Hollingshead, A. (1994), *Groups Interacting with Technology*, Sage Publications, San Fransisco, CA.

McLeod, P. (1992), “An assessment of the experimental literature on electronic support of group work: results of a meta-analysis”, *Human-Computer Interaction*, Vol. 7 No. 4, pp. 257-80.

McLeod, P. and Liker, J. (1992), “Electronic meeting systems: evidence from a low structure environment”, *Information Systems Research*,Vol. 3 No. 3, pp. 195-223.

Miles, M. and Huberman, A. (1994), *Qualitative Data Analysis*, Sage Publications, Thousand Oaks, CA.

Miller, S. (1982), “Quality and quantity: another view of analytic induction as a research technique”, *Quality and Quantity*, Vol. 16 No. 2, pp. 281-95.

Olson, J., Olson, G., Storrosten, M. and Carter, M. (1993), “Groupwork close up: a comparison of the group process with and without a simple group editor”, *ACM Transactions on Office Information Systems*, Vol. 11 No. 4, pp. 321-48.

Orlikowski, W. (1992), “The duality of technology: rethinking the concept of technology in organizations”, *Organization Science*, Vol. 3 No. 3, pp. 398-427.

Orlikowski, W. (1993), “CASE tools as organizational change: investigating incremental and radical changes in systems development”, *MIS Quarterly*, Vol. 18 No. 3, pp. 309-40.

Orlikowski, W. and Robey, D. (1991), “Information technology and the structuring of organizations, *Information Systems Research*, Vol. 2 No. 2, pp. 143-69.

Pondy, L. (1967), “Organizational conflict: concepts and models”, *Administrative Science Quarterly*, Vol. 12 No. 3, pp. 296-320.

Robey, D. (1984), “Conflict models for implementation research”, in Schultz, R. and Ginzberg, M, (Eds), *Applications of Management Science*, JAI Press, Greenwich, CT.

Sachs, P. (1995), “Transforming work: collaboration, learning, and design”, *Communications of the ACM*, Vol. 38 No. 9, pp. 36-46.

Seidler, J. (1974), “On using informants: a technique for collecting quantitative data and controlling measurement error in organization analysis”, *American Sociological Review*, Vol. 39 No. 12, pp. 816-31.

Suchman, L. (1983), “Office procedures as practical action: models of work and system design”, *ACM Transactions on Office Information Systems*, Vol. 1 No. 4, pp. 320-8.

Suchman, L. (1995), “Making work visible’, *Communications of the ACM*, Vol. 38 No. 9, pp. 56-65.

Thomas, K. (1975), “Conflict and conflict management”, in Dunnette, M. (Ed.), *Handbook of Industrial Psychology*, Rand-McNally, Chicago, IL.

Tyran, C., George, J. and Nunamaker, J. (1993), “Group support for technical review teams: an exploratory investigation”, *Proceedings of the 26th Hawaii International Conference on Systems Science.*

Walz, D., Elam. J. and Curtis, B. (1993), “The dual role of conflict in group software requirements and design activities”, *Communications of the ACM*, Vol. 36 No. 10, pp. 63-76.

Weick, K. (1995), *Sensemaking in Organizations*, Sage Publications, Thousand Oaks, CA.

Weick, K. and Roberts, K. (1993), “Collective mind in organizations: heedful interrelating on flight decks”, *Administrative Science Quarterly*, Vol. 38 No. 5, pp. 357-81.