

**Conversational Coherence:
Using Email Threads to Coordinate Distributed Work**

JoAnne Yates
Sloan School of Management
MIT (E52-544)
50 Memorial Drive
Cambridge, MA 02142
(phone) 617-253-7157
(fax) 617-253-2660
jyates@mit.edu

Wanda J. Orlikowski
Sloan School of Management
MIT (E53-325)
50 Memorial Drive
Cambridge, MA 02142
(phone) 617-253-0443
(fax) 617-258-7579
wanda@mit.edu

Stephanie L. Woerner
Sloan School of Management
MIT (NE20-336)
50 Memorial Drive
Cambridge, MA 02142
(phone) 617-452-3222
(fax) 617-253-4424
woerner@mit.edu

November 27, 2006

Working Paper—Please do not cite without permission

Abstract

This paper explores the central role of conversational threads in facilitating the ongoing, distributed work of one virtual organization. In studying the electronic mail exchanges of organizational members during one year, we found that they engaged in a range of threading activity to establish and maintain continuity, coherence, and coordination in their collaborative work over time. In particular, we found that organizational members used threads in multiple ways: in isolation, to carry on *individual conversations* that focused their attention and action on a particular topic over a short period of time; in parallel, to carry on *concurrent conversations* that enabled their engagement with multiple topics during the same period of time; and in sequence, to conduct *recurrent conversations* that allowed them to connect related work over long periods of time. We conclude by discussing the implications of conversational threads for research on virtual organizing.

Key Words

conversational coherence; distributed work; email; threading; virtual teams; virtual work.

In recent years, we have seen considerable interest in the use of computer-mediated communication (CMC) to constitute virtual communities and teams. Whereas geographically co-located teams involved in collaborative work can use a variety of specialized standardized cooperative practices along with associated specialized artifacts to help coordinate themselves (Schmidt and Wagner, 2005), virtual teams have access to many fewer (if any) physical artifacts, and instead are likely to use forms of mediated communication for coordination. Experiences with various forms of mediated communication have raised important questions around how members of groups and organizations maintain conversational and interactional coherence electronically (Herring, 1999). In work settings, where groups must coordinate their actions through communication, several critical questions take on special force. What are effective means of coordinating work over time and across space? How do members of groups and organizations establish and maintain coherence and continuity in such distributed conditions? How do members use technology to mediate their distributed coordination? These are the questions we address in this paper.

Virtual communities that exist through such text-based CMC systems as bulletin boards, listservs, Usenet newsgroups, and Internet Relay Chat (IRC) have been the most studied in existing research as they offer easy access to data (e.g., see Herring 1999; Murray 2000; Hiltz and Turoff 1978; Rheingold 1993; Lewis and Knowles, 1997). On a smaller scale, virtual or computer-mediated teams composed of students assigned to specific teams for classroom-based tasks are accessible for experimental purposes and have also been widely studied (e.g., Sproull and Kiesler, 1986; Cornelius and Boos 2003; Walther, 2002; Garcia and Jacobs 1999). Similar interest in use of CMC in the work world—rather than recreation, hobbies, or education—has increasingly focused on virtual ways of organizing work within and across firms, including, for example, virtual project teams (Lipnack and Stamps, 1997; Majchrzak et al., 2000b; Maznevski

and Chudoba, 2000) and geographically-distributed occupational communities and organizations (Markus et al., 2000; Jones, 1997). Our research is situated in this world of work.

We report here on an empirical study that examined a small software development start-up—Little Company (LC)—whose five members were geographically dispersed in four different cities and three different time zones, and who relied to a great extent on electronic mail to conduct their work.¹ Our research uses the notion of conversational threads as an analytic lens to examine electronic mail exchanges among LC members as they engaged in their ongoing and distributed software development work. We follow McDaniel et al. (1996, p. 41) in defining a thread as ““a stream of conversation in which successive contributions continue a topic, following an initial contribution which introduces a new topic.”” In particular, we are interested in understanding how LC members maintained coherence in their ongoing email activity, that is, how these members engaged in different types of threading activity as a way of coordinating their ongoing work over time and across distance, and as a means of establishing and maintaining conversational coherence and continuity. As Nardi, et al. (2000, p. 79) note, current media theories tend to assume “that communication is best studied one interaction at a time, rather than in a temporal sequence spanning multiple discrete interactions.” The analytic lens of conversational threads allows us to study communication not in terms of discrete encounters but as embedded within the ongoing flows of activity constituting the work of the organization.

In the next section, we survey literature on conversation in CMC, and then on conversational coherence in CMC. We then discuss our site and methods, providing some background on the LC team, and describing the methods we used to analyze threading activity in its members’ electronic mail. In the following section we report our analysis of challenges to, and techniques for achieving, conversational coherence in three types of threading activity used by LC members to accomplish their virtual work. In particular, we examined the dynamics

within, across, and among conversational threads. Focusing within threads, we see that the members used individual conversational threads to carry on coherent conversational exchanges, typically within circumscribed time periods. In the face of challenges such as time span and disruptions in turn taking, they used a range of techniques (e.g., subject line repetition and migration, lexical repetition, embedded messages) for maintaining coherence. Looking across threads, we see that these members often used parallel threads to carry on multiple concurrent, temporally interleaved conversations on different topics, while keeping each conversation distinct for coherence. They addressed the additional challenges of conversational number and complexity by using additional techniques such as thread convergence and divergence. Looking among conversational threads, we see that they used multiple threads in sequence over time to conduct recurrent conversations on complex and recurring issues or tasks that were crucial to the team's work. We also found that LC members addressed the additional challenges of extended time span and high uncertainty by leveraging textual persistence and provisional settlements (Girard and Stark, 2002). We conclude with a discussion of the role that conversational threading activity plays in coordinating distributed activities over time and across distance.

1. Literature on Conversational Coherence in CMC

Since the mid-1980s, academics in fields ranging from computer science and linguistics to computer supported cooperative work (CSCW) have looked at language, discourse, and interaction through CMC, with a range of findings around its linguistic and conversational features. Some of these results were seemingly contradictory. For example, some of the early papers looked at the presence of "flaming," or uninhibited language over electronic media (Sproull & Kiesler, 1986, 1991; Kiesler et al., 1984), while others found a lack of such language (e.g., Yates and Orlikowski, 1993; Markus, 1994). Differences in the electronic media

themselves, as well as in contexts of use and users' purposes (zero-history lab groups vs. ongoing organizationally embedded groups with tasks to perform), all contributed to the diversity of findings.

The term CMC has been used to refer to a large range of electronic media with quite different characteristics. In addressing the question "What is CMC?" Murray (2000) traces the term to Hiltz and Turoff's (1978) early study of computer conferencing, but goes on to note that it has been used by a variety of researchers to refer to studies of email, discussion lists, Internet Relay Chat (IRC), bulletin boards, chat rooms, MUDs (Multi-User Dungeons and Dragons), and even the World Wide Web (WWW). She and others (e.g., Herring, 1999) reviewing this literature restrict the term slightly by referring to "text-only CMC," thus eliminating the Web, videoconferencing, and other multi-media channels. As both Murray (2000) and Herring (1999) point out, however, even when restricted to text, CMC still includes a broad range of media with quite different characteristics.

Within that broad range, a distinction is frequently made between "real time" and "non-real time" interaction (Black et al., 1983), or between synchronous and asynchronous modes of CMC (Chang, 2005; Murray, 2000). Interestingly, although Murray (2000) uses synchronous and asynchronous to refer to modes of communication, she (as well as many others) identifies these modes with media (the asynchronous mode with bulletin boards, email, and discussion lists; and the synchronous mode with IRC, Instant Messaging, and chat rooms). Nevertheless, she also notes that the seemingly binary distinction can blur, as when email is used nearly synchronously and when system lag makes chat not quite synchronous. Indeed, some researchers refer to text-based communication such as instant messaging, where a participant composes a whole message before sending/revealing it to other participants, not as synchronous, but as "quasi-synchronous CMC"

(e.g., Baym, 1996; Garcia and Jacobs, 1999). Generalizations about any characteristics of CMC are problematic because they ignore how the technology is being used.

A second set of factors that have shaped findings in studies of CMC are contextual. For example, we can look at one cluster of contextual factors that includes the size, boundedness, and general purpose of CMC interactions among a set of participants. Some studies have focused on mass interaction in computer-mediated virtual communities or virtual publics (Jones et al., 2001; Rheingold, 1993). Many such communities are defined at least initially by common social, recreational, or informational interests, as in the various Usenet lists. Further distinctions can be made among such lists on the basis of whether these interests are occupational, recreational, and health-related (Galegher et al., 1998). Baym (1996) notes that such discussion lists are “a hybrid between mass and interpersonal communication,” (p. 315) as well as using language that is a hybrid of written and oral communication (Ferrara et al., 1991; see also Murray, 1985, 1988). In such large virtual communities, where posters and lurkers come and go, membership is a loosely defined term. More recently, some scholars have begun examining the electronic communication that takes place within large open source communities, which are more task-oriented (e.g., see Lanzara and Morner, 2004), though many have the loosely defined, voluntary membership and minimal authority structure typical of Usenet communities.

Other studies have focused on the use of CMC in small group communication, often around specific purposes such as making a decision or performing a task, rather than interacting with no common goal, as in most Usenet groups. These small groups, generally clearly bounded, form a very different context from large discussion groups, and a context more relevant to the small, task-oriented LC team studied in this paper. Many studies of small groups or teams have been conducted in a lab or quasi-lab setting, with student pairs or teams as subjects (e.g., Baron, 2004; Condon and Cech, 1996; Cornelius and Boos, 2003). Studies of history-less teams in lab settings

highlight important language and communication issues (e.g., flaming), but have difficulty exploring how CMC is used to connect ongoing, organizationally situated teams and pairs. More recently, some lab studies of virtual student teams have offered interesting insights through following teams over a longer period of time (e.g., Cramton, 2001) and more interactions (Walther, 2002).

Field studies of the dynamics of real communication among individuals in ongoing groups or teams using CMC are relatively rare, since access to *in situ*, ongoing communication in non-student teams is difficult to gain. A few researchers have studied geographically distributed, task-based teams using a variety of media (e.g., Hinds and Mortensen, 2005; Maznevski and Chudoba, 2000; Rennecker, 2001) and have highlighted the importance of such issues as conflict and context. More studies have focused primarily on use of text-based CMC such as email lists (e.g., Orlikowski and Yates, 1994) or groupware tools (e.g., Yates et al., 1999; Yates et al., 1996), examining the genres and genre systems enacted by participants.

As this discussion makes clear, the literature on CMC looks at a variety of specific media, and studies of use take place in a variety of different settings, from the lab to Usenet groups to geographically distributed task teams. We now turn to assessing what is known about conversational coherence, designated as “interactional coherence” by Herring (1999), in CMC. Some of the relevant literature focuses on designing technical means of increasing coherence and decreasing cognitive overload (e.g., Donath et al., 1999; Jones et al., 2001; Lewis and Knowles, 1997; Severinson-Eklundh and Macdonald, 1994), and some focuses on understanding how humans create coherence in computer-mediated conversations without technical aids (e.g., Black et al., 1983; McDaniel et al., 1996; Orlikowski and Yates, 1994; Reed, 2001). Nonetheless, this literature provides valuable insights into a number of coherence mechanisms.

Studies focused on both technical and human aspects of CMC tend to assume the importance of conversational threads in achieving coherence. The use of conversational threads (a term adopted from analysis of pre-electronic written communication) to obtain topical coherence is common in both online and offline communications (Black et al., 1983; McDaniel et al., 1996; Donath et al., 1999; Herring, 1999; Millen and Patterson, 2002). Online threads have been defined in multiple ways, including some that are dependent on technical features of the CMC system such as the reply function and subject-line field in messages (see Lewis and Knowles, 1997). We follow McDaniel et al.'s (1996) communicative definition of threads as streams of conversations where successive contributions continue a topic. In a similar focus, Lanzara and Morner (2004) note that threads “act as focal points” (p. 3) in the communication of open source groups, and that their “next-next-next” sequential pattern or “close sequencing” fosters “inter-temporal connectivity” (p. 4).

Identifying and untangling such conversational threads is frequently difficult, however, in CMC exchanges. Hutchby (2001, pp. 181-191) examines some of the difficulties of and mechanisms for untangling the quasi-synchronous conversations taking place in public Internet Relay Chat (IRC) discussions with many participants (often unknown to each other). Because IRC technology does not support subject lines and “in reply to” or “reference” fields, recreational IRC users either enjoy the ambiguity in conversational sequence or use techniques such as naming the person they are responding to in order to reduce it. In email and listserves, subject lines and “in reply to” or “reference” fields are a starting point in identifying topical threads. Lewis and Knowles (1997) note the inherent difficulties of tracing threads of conversation by these means, including the fact that people often reply to a previous message only to retain the address, making the “in reply to” or “reference” field in some CMC systems potentially misleading. In attempting to improve the design of interfaces to reduce cognitive

overload on users in large groups such as Usenet newsgroups, they test several possible technical strategies for matching text to create conversational threads automatically (e.g., special software using subject lines, internal quotes, and lexical repetition). Still, purely technical solutions to this difficulty are hard to achieve.

Only a few studies have looked at threading activity systematically. In comparing the threading activity of face-to-face and computer-mediated communications, McDaniel et al. (1996) find more threading by those participants engaged in computer-mediated interactions. In addition, they report that the threading activity of computer-mediated communications includes more concurrency (defined as the interleaving of contributions from different threads) than that of face-to-face interactions, indicating that users of electronic media tended to participate in multiple conversational threads at the same time (see also Whittaker et al., 2002). The notion of concurrency in electronic communication resembles the “crisscrossing of communicative chains” identified by Reder and Schwab (1990, p.12), where they argue “the many switches and intertwinings of task and channel ... are fundamental features of workplace communication.” In studying CMC in two open source email lists, Lanzara and Morner (2004, p. 14) find that long threads are in the minority, with 33% and 47% of potentially thread-initiating messages in the two communities never receiving any responses (cases they consider threads of length one), while another 44% and 34% of threads have only two to four messages. Both thread concurrency and the total number of messages and threads clearly pose challenges to the conversational coherence potentially provided by threads.

A few researchers have noted other, non-technical mechanisms through which participants create conversational coherence, focusing on combinations of social norms and adaptation around technical defaults. In a study of the early 1980s email list communication of a moderately large and loosely defined group of artificial intelligence language designers

developing the Common Lisp computer language, Orlikowski and Yates (1994) identified the dialogue genre by its use of embedded messages, that is, pieces of (or entire) previous messages included and replied to within subsequent messages of a thread. This group of early email users increased their use of the genre over the period studied, indicating a growing norm around this means of maintaining conversational coherence. In their study of open source communities Lanzara and Morner (2004) referred to a similar device as “nested conversations,” emphasizing the role of nesting in creating a coherent conversation. In another study of Internet newsgroup interaction, Reed (2001) found that participants used several practices to maintain what he terms “sequential integrity” in conversations, thus reducing the burden of untangling threads. These practices, including quoting (similar to embedding messages) and other editing techniques, reflect norms developed over time by the newsgroup participants to help create conversational coherence.

A very early paper on CMC (Black et al., 1983) that reported a study of communication in two different electronic media argued that different coherence mechanisms were used in real and non-real time interactions. The authors argued that “Non-real time discussions, unlike real time discussions, employ multiple discourse threads” (p. 70) during the same time period, requiring the reader to unravel the threads to restore sequentiality. Their finding that multiple threads were used simultaneously in non-real time conversations, which they linked to temporal delays in the medium, is intriguing, but their examples came from an educational context, with asynchronous instructional interaction between students and professor over an electronic messaging system called MSG, and synchronous “helping dialogs” between research assistants and students over the TENEX system. In both these cases the interactions were shaped by the relationships and purposes enacted in the educational context, as well as by the relative

inexperience of the student subjects (who were trained in using the TENEX system but had not necessarily used it previously) and the specifics of the two systems.

Thus far, the CMC literature provides valuable but partial views of how teams create conversational coherence with various technologies in online communication. Our study of the mediated communication of a small software development company over time is intended to help address this gap in the literature.

2. Site and Methods

LittleCompany (LC) is a small, primarily self-funded software start-up company building a programming language product, the LC system. During the time period studied, three of the members (Dan, Keith, and Robert) worked full time and two (Martin and Fred) worked part time (the two part-time members eventually dropped out, though after the time period covered in this analysis). Members of the company were geographically dispersed; the five original members worked in four different cities and three different time zones across the U.S. This virtual company had no central physical location, as the members worked out of their homes or private offices. Although they were geographically dispersed, there were multiple pre-existing interpersonal ties among the members: three of them—Dan, Keith and Fred—had received their computer science doctoral degrees at the same university; Dan and Keith were friends and had written papers together; Keith and Robert were friends and neighbors who worked together in a large company before LC was started and Robert moved to a different city; and Fred and Martin, who had worked together many years previously, were friends who lived in the same area and met regularly to play racquetball. The work of this company was done in collaboration among all five members, but the three full-time members did most of the coding while the two part-timers were more involved in the business aspects of the start-up. The primary form of communication was

electronic mail, supported by frequent dyadic phone calls, weekly group phone meetings, and a few dyadic (but no all-LC) face-to-face meetings.² During this period, DSL service was not yet available, and as dial-up service was charged by the hour, LC members checked and downloaded email frequently but were not continuously connected.

For this study, we analyzed primarily the electronic mail archives belonging to one of the full-time members of LC who had saved all of his email dating from the inception of the company. This member also served as a key informant during our study. At the final stage of analysis, we added a few messages from the partial email archives of the other two full-time members (see below). Our analysis of email archives starts in early 1997, when all members of the organization were in place and work on the LC system product began, and ends at the alpha release of the product, approximately one year later. Because email messages served as the central source of data for this paper, our method may be said to fit Herring's (2004, p. 339) definition of computer-mediated discourse analysis, which, "at its core is the analysis of logs of verbal interaction (characters, words, utterances, messages, exchanged, threads, archives, etc.)"

Our analysis began with all messages in our informant's archive related to the company, its members, and the product — a set consisting of 3,641 messages. In order to identify threads, we first cleaned the date and time data. As the email messages covered three time zones, we found that we could not use the local time stamped on each message to order the messages into coherent threads. We thus converted the local time stamps to Greenwich Mean Time (GMT), which standardized the time each message was sent.

Having selected our sample and organized the email messages into chronological order, we operationalized the key concept of threads (Herring, 2004, p. 354). Acutely aware of the difficulties of using only subject lines in defining threads (Lewis and Knowles, 1997), we followed a multi-stage, predominantly manual process. We first grouped the email messages

into provisional thread clusters by matching subject lines, then deleted duplicate messages. We occasionally grouped together messages that were chronologically close to each other with subject lines that didn't match exactly but which seemed related. This process produced an initial set of 626 threads consisting of 2,215 email messages.

We next read through the subject lines of the initial threads, flagging messages that had been provisionally but questionably included in a thread, indicating places where an initial or interim message was missing in an exchange, and noting possible relationships among separate threads. We then went back to the email messages themselves, using close reading as the basis for adding messages to, subtracting messages from, and combining threads. In cases where the technical content made it difficult to judge whether a message belonged to a thread, we consulted with our key informant. At this stage, we had identified 560 threads composed of 2,431 messages, indicating that well over one half of the email messages we had analyzed could be clustered into distinguishable threads.³

Next we eliminated threads involving just two of the members, because our data, while comprised primarily of messages to all LC members, also included two-party messages between our key informant and other individual members of LC, while excluding two-party messages solely between any two other LC members.⁴ To avoid skewing the data, we used only threads involving all three full-time members of LC at some point in the exchange, whether as message senders or recipients. Thus, we retained threads that included some two-party messages interspersed with the messages to all three full-time LC members. This process left us with 469 threads with 2,163 messages.

In the next stage of the analysis, we focused on an interesting phenomenon that we had observed from our close reading of the threads: the presence of related but separate and distinct threads that constituted what we call recurrent conversational threads. To identify these, we

analyzed groups of related threads by first examining related subject lines, creating a subset of threads from the larger archive. Then we read the threads multiple times and consulted with the key informant, adding related threads and subtracting unrelated ones. Through this process we identified five sequences of threads that made up recurrent or ongoing conversations, designating them by their subjects (all technical terms for aspects of the system): *Linker*, *Make*, *BigNums*, *Hash*, and *GC*.⁵ To assure that we were not including threads that simply concerned the same topic but that were not really part of an ongoing conversation, the authors reread the five recurrent conversational threads (each author taking primary responsibility for one or two of them), retaining only those threads that were part of the ongoing conversations. In any ambiguous cases, all three researchers read the thread and discussed it until they reached agreement. In several cases, the authors decided that two threads should be combined into one because of the close linkage and ongoing nature of the conversation. This analysis resulted in a total of 89 threads in 5 recurrent conversational threads.⁶ For these recurrent conversational threads, we also identified “missing” messages that did not exist independently in the primary email files, but pieces of which appeared in other messages (these missing messages were almost exclusively between the two full-timers other than our informant). We then sought these missing messages in the partial files of the other two full-time LC members, inserting them when we found them to make our recurrent conversational threads more complete.⁷ Each of these recurrent conversational threads was further analyzed to classify whether it appeared to be settled, provisionally settled, or unsettled (at least in the email data we had). Each thread was discussed by all three authors until agreement on the coding was reached.

The following section presents the results of our analyses. When we quote from the LC email messages, we have changed wording and details as needed to disguise the company’s

identity and technology. We have also cleaned up obvious typos to aid readability but retained the typing conventions of the participants (e.g., some members rarely capitalized).

3. Conversational Coherence in LC Communication

Conversational exchanges, or threads, constitute the building blocks of our analysis of the LC electronic archive, and the most basic device providing conversational coherence. We examined conversational coherence at three different levels—*within* threads, *across* threads, and *among* threads—identifying and characterizing three types of threading activity used by the LC members to structure their ongoing work and interaction over time and space, as well as issues around achieving conversational coherence at each level. First, LC members used *individual conversational threads*, or conversational exchanges, as a way to provide coherence to their asynchronous interactions. As the key informant noted: “Whenever we had a sequencing thing, you would see a thread. We explicitly did it in an email because phone calls evaporate.” These threads are tightly-coupled contributions on a single topic that resemble those identified in prior studies of electronic interaction (McDaniel et al., 1996; Donath et al., 1999). Second, LC members used threads in parallel to allow conversations on separate topics to be interleaved over the same general time period, resulting in *concurrent conversational threads*. Third, they used threads in loosely coupled sequences we call a *recurrent conversational threads* to conduct ongoing conversations on complex topics over long periods of time.⁸ While using threads in parallel allowed the LC members to conduct work on multiple fronts at the same time, using them in sequence as part of recurrent conversations allowed them to pick up and continue work on a particular front after some pause or provisional settlement had halted activity for a time. We examine challenges to coherence and techniques for addressing the challenges at each of these three levels below.

3.1 Individual Conversational Threads

We identified 469 threads in the data (see Table 1). These threads averaged 4.6 messages per thread, with a median of 3 messages per thread. Figure 1 shows the length and distribution of threads over time. Most of the threads we identified consisted of only 2 or 3 messages, with 38% of the total consisting of 2 messages and another 22% of 3.⁹ These brief threads typically included an inquiry and a response, with sometimes an additional comment from the originator of the thread or occasionally from a third participant. The default subject line in the second and third messages (if the respondents simply hit reply and didn't alter the subject line) was "Re:" plus the subject line of the original message, and the initial message would be reproduced, either in its entirety with the response following it, or in pieces with specific responses to it. Typical 2- and 3-message threads were concluded within a single day, with an average elapsed time of 11 hours and 28 minutes, and a median elapsed time of 3 hours and 20 minutes. The shortest such interaction took less than one minute, indicating that the recipient of the original message was online and replied immediately to it. Such cases indicate almost synchronous interactions. More typically, however, the short, 2-3 message interactions spanned several hours and were clearly asynchronous.

Many individual conversational threads concerned detailed technical issues, and the messages often contained pieces of code or stack traces from a system crash. Other exchanges involved administrative (e.g., internal coordination) or business issues. In the following example, a 3-message thread is used to coordinate a change in time for a regular phone conference:

Date: Sun, 14 Sep 97 15:22

From: Fred

To: all

Subject: next week's conference call

Gentlemen:

Since Martin is unavailable on Wednesday at the usual time I propose that we move the weekly conference call to Tuesday at (1500/1300/1200) hours instead. This does not conflict with any of the constraints that people have noted in email and in telephone conversations with Keith, Martin, and Dan they have said that this was an acceptable time for them.

Please send email around indicating your agreement or veto.

I have two agenda items for the meeting:

** patent issues: choice and procedure*

** foreign-function strategy*

regards..fred

--

Date: Mon, 15 Sep 1997 13:42

From: Robert

To: Fred

Cc: all

Subject: Re: next week's conference call

Fred wrote:

>

> Gentlemen:

>

> Since Martin is unavailable on Wednesday at the usual time I propose that we move the weekly conference call to Tuesday at (1500/1300/1200) hours instead.

This is okay with me.

--

Robert

To: all

From: Dan

Subject: Re: next week's conference call

At 01:42 PM 9/15/97, Robert wrote:

>Fred wrote:

>>

>> Gentlemen:

>>

>> Since Martin is unavailable on Wednesday at the usual time I propose that we move the weekly conference call to Tuesday at (1500/1300/1200) hours instead.

>

>This is okay with me.

Works for me too.

Dan

While the majority of threads were short, a few were quite lengthy. Of the 469 threads, 49 (10%) included 10 or more messages, and 10 threads (2%) included 20 or more messages. In these longer threads, which included as many as 32 messages, LC members thrashed out company strategy, identified and fixed bugs in the LC system, or aired disagreements about aspects of the LC system. The average elapsed time for all threads was a little over one day, indicating that interactions, even long ones, were typically completed rapidly, a reflection of the fact that the three key participants worked full time for LC.¹⁰ One 32-message thread concerning a system

crash, for example, took place over a period of two and a half days. During this time, the three full-time LC members ran various traces to diagnose the problem, identified a “smoking gun,” fixed the bug, tested the fix, and committed new code to the site of the LC system.

In general, LC members used individual conversational threads to initiate, conduct, and close a conversation. In some cases, conversations were not completed within the email thread, and references suggest that the participants switched to telephone.¹¹ In many cases, however, the interaction may be traced from beginning to end in the email thread.

Although conversational threads help tie together asynchronous communicative interactions, LC participants still faced challenges or potential challenges to conversational coherence in a given thread, in comparison to synchronous face-to-face or phone conversations. First, conversational threads in LC’s primarily asynchronous interactions spanned time; even if the time span was relatively short within a given thread, it was a period during which the participants engaged in other work, non-work activity, and other interactions, whether face to face or through other media. The lag between conversational turns could lead to loss of focus and coherence, as well as possibly contributing to topic decay or drift when conversational participants digress from the topic initiated. Such decay has been observed in studies of CMC, especially in those of social, quasi-synchronous IRC interactions (see Herring, 1999, for a review of such studies). Another major challenge to conversational coherence in an email conversation is what Herring (1999) refers to as disrupted turn adjacency, which occurs when responses aren’t received immediately after the message to which they refer, but are interrupted by messages on other topics (see also the next section on concurrent conversational threads) or from other participants. Below is a 30-minute sample of the company in-box of our informant:

<i>From</i>	<i>To</i>	<i>Time</i>	<i>Subject Line</i>
<i>Dan</i>	<i>Dan, Keith, Fred, Robert</i>	<i>2:10</i>	<i>What's this mean?</i>

<i>Dan</i>	<i>Dan, Keith, Fred, Robert</i>	<i>2:22</i>	<i>Re: linker bugs try 2</i>
<i>Robert</i>	<i>Dan, cc All</i>	<i>2:23</i>	<i>Re: What's this mean?</i>
<i>Keith</i>	<i>Dan, Robert, Martin</i>	<i>2:29</i>	<i>code size</i>
<i>Robert</i>	<i>Dan, cc All</i>	<i>2:32</i>	<i>Re: linker bugs try 2</i>
<i>Robert</i>	<i>Keith, cc All</i>	<i>2:34</i>	<i>Re: code size</i>
<i>Martin</i>	<i>All</i>	<i>2:36</i>	<i>Libraries and Performance Benchmarks</i>
<i>Keith</i>	<i>Dan, Robert, Martin</i>	<i>2:36</i>	<i>man in the middle attack</i>
<i>Keith</i>	<i>Dan, cc Keith, Fred, Robert</i>	<i>2:39</i>	<i>Re: linker bugs try 2</i>

In this half-hour time span, LC members are engaged in an ongoing conversation about linker bugs, start a new conversation about code size, receive an announcement about benchmarks, and get some information about an attack on computer security; these messages are interwoven together, appearing on the screen as they arrived in email, rather than being grouped thematically. Thus Robert’s 2:23 reply to Dan’s 2:10 message with subject line “What’s this mean?” comes only after an intervening 2:22 message on a different issue, and similarly the 2:22, 2:32, and 2:39 contributions to the conversational thread on linker bugs are interspersed with messages on other subjects. To make it even more potentially confusing, Keith’s 2:39 message actually responds to a message sent by Dan over four hours previously (not included in the 30-minute sample shown), ignoring the intervening 2:22 or 2:32 messages in the thread. Such disruption of turn adjacency can lead to confusion and loss of conversational coherence.

In LC, several factors reinforced these challenges related to asynchronicity. Because LC members worked in three different time zones and had different individual work schedules (Im et al., 2005), a given participant might easily send an email when few or none of the others were online. Moreover, because they were using email with modem and phone lines during this first

year, they tended to use it in an asynchronous, batch-processing way. As our key informant told us in an interview,

When email was dial-up, you had to be connected to get mail. We used PPP. We had a certain number of hours of dial-up time free and had to pay for the rest. I would connect, get the email, and disconnect.

Batch processing of email and the differences in time zone and schedules only exacerbated the challenges to conversational coherence.

LC members used, whether explicitly or implicitly, several techniques that helped them maintain conversational coherence in spite of these challenges. The first technique was **limiting themselves to one work topic per thread**. In reading through the many messages in threads, we were surprised to see that the LC members seemed to limit their messages to a single work topic, and almost never drifted from that topic within a thread, whether defined solely by identical subject lines or also by close reading. LC members occasionally added personal notes to technical messages, or even focused primarily on personal, rather than technical or administrative, issues in a thread, but they rarely drifted from one work-related topic to another. For example, in a set of exchanges extracted from the archive before we grouped messages into threads, we see how LC members coped with new information, problems, or suggestions without creating topic drift.

<i>From</i>	<i>To</i>	<i>Subject Line</i>	<i>Message Summary</i>
<i>Keith</i>	<i>Dan, Robert, Martin</i>	<i>I think that the ball is now in robert's court</i>	<i>"I fixed the bug. It looks like I have another error. Here's how to reproduce it."</i>
<i>Robert</i>	<i>Keith, cc All</i>	<i>Re: I think that the ball is now in robert's court</i>	<i>"I fixed that problem but now I have another."</i>

<i>Robert</i>	<i>All</i>	<i>ISP shmISP</i>	<i>"I can't reach the ISP so I'll check-in tomorrow."</i>
<i>Keith</i>	<i>Robert</i>	<i>Re: ISP shmISP</i>	<i>"The ISP crashed last night. It's up now."</i>
<i>Keith</i>	<i>All</i>	<i>Re: I think that the ball is now in robert's court</i>	<i>"It looks like it's my problem now."</i>

Keith and Robert have an exchange about a bug but when Robert has to tell Keith that he can't check in the bug fix that he discussed in the second message, he creates a new thread to update Keith on why he can't check in. Keith responds to each of those messages separately, maintaining the "one work topic to a thread" convention.

When we asked our key informant whether that pattern was an explicit policy, he replied as follows:

I don't think we had a rule...[but] if you mix things up, well then they're mixed up. That's a tautology, but...

This strong logical norm might have been inculcated in their computer science education or perhaps imported from previous jobs in software development, but it prevented drift or decay in work topic within given messages or threads.

Conversational coherence was also maintained by using **repeated or migrating subject lines**. The default repetition of the subject line worked well to indicate what thread a given message belonged to since they avoided topic drift. LC members often changed the default subject line, however, to increase coherence by signaling the message's role and place within an exchange. For example, the subject line could indicate progress in a problem solving interaction--*"very close (ball between Utah and New York)"* (the locations of two of the LC members at that time); coordinate whose turn it was to work on the focal technical problem of a thread--*"ball*

in robert's court"; announce the nature of a problem already signaled by another-- "*yes we have bugs*"; acknowledge fault in an interaction-- "*my turn to plead insanity*"; or to express humor or word play in the midst of problem solving—"more march makefile mystery" (makefile was the subject of the ongoing problem-solving interaction, which was taking place in March). One long thread, for example, started out with a subject heading of *Symbol file format* (an aspect of the system's Linker). Soon that subject line migrated through the following, among others:

Re: Symbol file format

Symbol file format, round 2

Linking, take 3

Linking, take 4

Son of linker, part 5

Return of son of linker (#7)

Here, the LC participants were expressing their sense of how long it was taking to solve the problem, but doing so humorously and in a way that made it clear that the thread was continuing the same topic, even though the subject line was changing.

Another frequently used technique for achieving conversational coherence, **embedded messages**, involved including previous messages or pieces of messages, in indented format, into a new message. Like Orlikowski and Yates's (1994) embedded messages and Lanzara and Morner's (2004) nested conversations, these indented pieces of previous messages showed two (or more) conversational turns at once, strengthening the connection of the current message to the previous one(s). In the thread used to schedule a phone meeting, shown above, we can see that Robert embedded the relevant piece of Fred's message and Dan responded to Robert's message by embedding it (including its piece of Fred's message) before adding his reply. This technique allows them to respond briefly while maintaining the integrity of the conversational exchange and making the structure of the conversation visually clear (easy to do since the email

software provided the previous message already indented in the reply). In the 30-minute in-box sample shown above, it also prevents the confusion that would otherwise result from Keith's delayed response to Dan's much earlier message in the "linker bugs try 2" thread, making clear which message he was responding to in his 2:39 message.

Another technique, **lexical repetition** (Lewis and Knowles, 1997), involved repeating key words from previous messages to emphasize the continuity of the conversation. In many cases, the repeated words were technical terms (as the subject lines with plays on *linker* demonstrate). The following brief thread, built on an announcement followed up by joking and word play (here the thread centered on the humor, not the work message, in the initial message), illustrates both embedding and lexical repetition:

To: all

From: Dan

Subject: fyi

In case anyone is wondering, we just had a shitpile of snow. Keith's power is out, and my ISP is unusually hard to reach.

Dan

Sent after Dan talked to Keith (who lived in the same time zone, though hundreds of miles away) by telephone, this message informed the other LC members of the weather in their part of the country, indirectly explaining that Keith would be unreachable by email, and Dan only sporadically reachable. No work-related response to this message was needed, but Robert responded humorously, embedding the previous message and posing a question about LC policy around snow days (a joke because LC was a tiny, self-funded start-up):

From: Robert

To: all

Subject: Re: fyi

Dan wrote:

> In case anyone is wondering, we just had a shitpile of snow. Keith's power is out, and my ISP is unusually hard to reach.

Does LittleCompany have snow days?

--

robert

Dan replied to Robert's joking question about snow days, which he embedded in his message. Then, after a passing comment on the snow's effect on his work, he continued in a humorous vein by repeating (in quotes to make the lexical repetition obvious) and pretending to define a colorful slang word from his original message (shitpile) as if it were a technical term in computer science:

To: all

From: Dan

Subject: Re: fyi

At 10:26 AM 4/1/97, Robert wrote:

>Does LittleCompany have snow days?

Don't think so. It picked a good day to snow; I made a couple of small fixes to instruction-form (form = reg/reg vs reg/spill vs reg/mem) selection, and have been testing them while I shoveled snow.

For reference, a "shitpile" is somewhere between 18 and 24 inches, packed. A pine tree in the yard next door broke off about 12 feet off the ground, an arborvitae in another neighbor's yard removed itself, roots and all, from the ground. Down the street there's a maple tree with two big branches down, across the front walk of a guy who's about 80. But, lucky us, all our services come in below

ground. Tonight it is supposed to get to 20, but tomorrow and the next day it is supposed to be warm.

Dan

In this message, then, both an embedded message and lexical repetition reinforced conversational coherence.

Herring (1999, p. 17) defines another factor enabling conversational coherence, which she calls “conversational persistence,” as “the availability of a persistent textual record of the interaction” (Herring 1999, p. 17). While this **textual persistence** is fairly brief when interactive chat systems (e.g., instant messaging) are used without archiving, email text persists until the message is deleted. Given the temporal length of most LC threads, LC members only had to save a message in the inbox for a single day to allow the receiver to go back and consult it in reading or writing subsequent messages. Taking advantage of textual persistence to do so reinforced other techniques in achieving conversational coherence.

One final set of techniques used to improve conversational coherence primarily in longer threads was composed of **summaries and closings**. In a thread concerning two bugs—Martin’s bug and test28 bug—and including 17 messages over one week, Dan sent the following message as the 14th of the sequence:

To: all@LC.com, dan@home.com

From: Dan

Subject: Martin's bug, test28 bug

Summary: they have a common cause, namely a missing line in the GC, which I fixed. However, I thought I'd record how I did this, so that someone else might be able to do this other than me.

...

He thus summarized what the group had discovered during the week and what he had done to fix the problem. The rest of the long message gave a detailed account of how he figured that out. This summary reminded LC members of what had linked the thread over the past week, making sure that the conversation was still coherent to all involved. Three subsequent messages closed the conversational loop on other aspects of the issue.

In addition to such internal summaries, longer threads typically ended with a statement of at least provisional closure. The final message in one very long (32-message) thread, for example, began with this statement from Robert to the group:

I have hopes that this is the last linker spec for a while. Changes are pretty minor, keith is getting tired of arguing.

Such statements reinforced the coherence and shape of the entire conversation by referring both to multiple iterations of the linker specifications throughout the thread and to the increasingly minor changes and reduced arguments characterizing recent messages in the thread.

Because LC members typically turned to email to discuss any issues that needed careful documentation, individual email threads formed the most basic conversational unit in the electronic mail exchanges of LC members. To overcome the challenges to conversational coherence posed by extended time spans and disrupted turn adjacency—challenges reinforced by their different time zones and work schedules and by their asynchronous mode of using email—they used several techniques to provide unity and coherence within the threads: restricting messages and threads to a single work-related subject, repeating or migrating subject lines through threads to signal coherence and progress, using embedded messages or lexical repetition to reinforce connections, taking advantage of textual persistence to make connections, and using summaries and closing statements in longer threads to remind LC members what the thread was about and what it had achieved.

3.2 Concurrent Conversational Threads

In addition to looking within threads, we also examined how LC members achieved coherence *across* conversational threads. Given the complexity of the work engaged in by the LC members, it is not surprising that they dealt with multiple subjects during the same general time period. We have noted that LC members tended to keep work topics in separate but parallel threads. We defined threads as being used *concurrently* when messages from one thread were temporally interleaved with messages from another. Concurrent conversational threads were the norm in LC, with 88.5% of all threads overlapping temporally with at least one other thread.

The fact that threads were used so infrequently in isolation both reflects and highlights the asynchronicity of LC members' email use and the complex web of interactions that allowed the group to do its work. It was common for other subjects and issues to arise before a given interaction was complete, leading to the interleaving of conversations. The median number of concurrent conversational threads that LC members participated in at a given time was three, and at some points up to seven threads ran concurrently (see Table 2 for more details). These concurrent conversational threads allowed the LC members to carry on parallel discussions about a variety of issues without having them become confused. Figure 2 shows how four threads interlinked during one particularly busy 5-day period.

The presence of multiple concurrent conversational threads at any given time posed additional cognitive challenges to LC members, beyond those already present within specific threads. Simply the increased number of conversations posed difficulties, since typically people engage in one oral conversation at a time, only rarely juggling two or more at the same time (Reinsch et al., 2006). Thus LC members had to keep the progress of all conversations going on at a given time adequately in mind to follow and contribute to them. In addition, with concurrency came added levels of complexity. Indeed, in some cases during the bug fixing stages a computer

crash would trigger an investigation, but it was not initially clear how many bugs or problems were involved, and thus how many discrete conversations were needed.

To help with the added problems created by the number and complexity of concurrent threads, LC members used two more techniques for achieving conversational coherence: thread convergence to reduce the number of threads, and thread divergence to decrease complexity.¹² Figure 3 shows an example in which three initially separate threads converged into a single thread. On Sunday, Keith sent a standardized notification that he had mounted a new version of the LC system onto the server. Initially, this message received no replies, nor did such messages normally. But on Tuesday two seemingly separate problems arose: Keith reported “cvs problems” and Dan reported “trouble linking test5.” Robert, who was responsible for issues related to the linker, replied to both Keith’s and Dan’s messages at once, combining both of the original subject lines into “re: trouble linking test5 & cvs problems,” to indicate that the two issues were related. Less than three hours later, after two more responses to this combined thread, Keith realized that the two problems were both related to the new version he had mounted on the server on Sunday. By Wednesday, Keith had replaced the faulty version with yet another new version, and Robert had successfully checked in some changes to the linker to work with this new version. Thus they combined what started as three separate threads into a single thread, reducing the number of concurrent conversational threads they were dealing with at that time.

Another technique LC members used to maintain conversational coherence between threads was thread divergence, or separating out individual strands of particularly complicated conversations in order to reduce the complexity of some conversational threads. Figure 4 shows a simplified diagram of thread divergence. Thread A began when Robert’s system got stuck and ultimately crashed as it tried to build the product. In the middle of work on the problem, Robert

posted what he thought was a fix, but soon discovered that it hadn't fixed the whole problem. At this point it became clear to him that there were actually two separate problems, so he separated out the conversation about the new compiler bug into a separate but concurrent conversational thread (Thread B). Because of the technical nature of the subject matter, we came to understand this relationship among the concurrent conversational threads only through iterative discussions with our key informant precipitated by our query about this flurry of messages.¹³

Thread concurrency aided conversational coherence across threads by separating discussions of different issues into separate threads. Although LC members considered multiple topics within a single synchronous meeting — as evident in the meeting agendas circulated before the weekly phone meetings and the notes taken on these meetings by one of the full-time LC members—they seem to have preferred separate, concurrent email threads to coordinate multiple topics in the asynchronous electronic medium. While no technical feature of the email prevented LC members from combining multiple topics within a single thread, the LC members were generally careful to discuss different issues, especially technical ones, in separate conversational threads. When desirable, they used the devices of thread divergence and convergence to reduce the number or complexity of conversations.

3.3 Recurrent Conversational Threads

In addition to looking within and across threads, we examined the interactions *among* threads over time in the LC data. In doing so we discovered an unexpected phenomenon, sequences of threads that connected related work over long periods of time, which we label *recurrent conversational threads*. These recurrent conversational threads were made up of multiple sequential threads (each of which was tightly linked internally) that were loosely connected to each other over time to make up a sporadic but ongoing conversation about a common task.¹⁴ As shown in Table 3, we identified 5 different recurrent conversational threads

in our data (each designated by the aspect of the LC system it concerned). These recurrent conversational threads included from 5 to 35 threads and from 25 to 280 total messages.¹⁵ These sequences of threads organized ongoing discussions about key technical aspects of the LC system that were central to LC's work and that recurred of necessity as the system was being developed.

When asked to reflect about what characterized this particular recurrent threading activity, our key informant explained:

There are four types or qualities of the things that we came back to again and again:

- 1. We didn't understand it in the beginning but we had to start. We would come back when we hit a problem.*
- 2. The problem is so hard that it isn't the most important thing to implement it correctly. You do a good enough job and then come back. You don't want to prematurely optimize because that pessimizes the use of your time. You want to see if it is good enough.*
- 3. The problems change, the specifications change, or the requirements change.*
- 4. Getting something right or changing it requires cooperation and coordination among everybody. ...*

We can see examples of each of these qualities in the longest recurrent conversational thread, which concerned building the system's *linker* (the module that links together all the other pieces of the large program). It extended over almost the entire period studied (see Figure 5), and, in fact, into subsequent years. The linker was a part of the LC system that had to be started and temporarily stabilized in order to make progress on other parts of the system (our informant's first criterion), as indicated in the following exchange:.

> - Does this really have enough power to express the things we want?

I think this is ok for now.

A few messages later in the same thread, Robert described a situation that the system might need to deal with in the future, and then explained:

While I don't want to do this right now, I'd like to avoid introducing stuff into the linker spec that makes this impossible.

This comment demonstrates the second criterion, the need to start somewhere and to keep options open.

The next several threads of the recurrent conversational thread occurred within the next few weeks, and started with questions about what the provisional *linker* specification implied for other parts of the system or for specific situations that might arise in the future, reflecting our informant's fourth criterion of the need for coordination among LC members. As is clear from Figure 5, *linker* was the subject of extensive discussion (and conflict) throughout March and April of 1997, continuing into May at a lower level. The subject lines on some of the threads revealed the unstable and evolving nature of the *linker*:

Re: linker versions

Re: getting things started

Re: one last time

Re: again

Even after this intense period of discussion ended, however, questions and requests for changes in the *linker* continued to emerge, triggered by clarifications of constraints imposed by the current form, other changes in the system, performance problems as the rest of the system got larger, and bugs revealed by all of the above. These factors are clearly part of our informant's third criterion. For example, in one thread Dan complained about performance problems, referring to the "obesity of our current system." After a query from Martin about why he considered it obese, Dan replied:

Because I run out of memory compiling some of our files, and we aren't even doing all of code generation, let alone optimization.

In a later thread, Keith made the following plea to Robert, who was the primary person responsible for the *linker*:

Robert, you have to understand that dan and I are dying here. We cannot make small changes and test things in a reasonable way. I spent almost all day yesterday waiting on links. This is a serious problem and we really need some relief.

Potential customers or markets also played a role in triggering recurrences of this ongoing conversation, though at this stage any customers were still prospective:

In the long run, this may require more recompilation than the customer is willing to deal with. If we try to back off to the point where we follow the ... rules at least to link time, we will give up some run time performance....

Dan and I have decided to know everything because this is the way to get some good performance in the short term. However, we are very cognizant of identifying these points so we will later be able to turn the knob back if the market dictates it.

If (When) we decide to turn that knob back, the linker will have to be enhanced because it is there where we will need to determine the object layout.

It clearly was not a surprise to LC members that the *linker* issue would keep coming up as the system, and the firm, evolved.

In addition to slow performance, bugs causing system errors or crashes also frequently triggered new threads of the *linker* recurrent conversational thread. Such threads would start by someone reporting an error message or system crash and asking for help in pinpointing the cause. In one case, for example, Dan named the function he had tried to execute in the subject line, then wrote as follows:

It does not work. After 14 minutes (elapsed) the interpreter running the linker falls over with an NT stack trace. I am going to retrofit some options for during on/off checking code, and I'm going to look into some ways that I might generate fewer relocations.

But, we've got a big problem.

As it turned out, this problem was solved relatively easily, but others were not. For example, Thread A of the five concurrent conversational threads shown in Figure 2 was triggered by a system crash during linking and was subsequently discovered to involve at least three bugs.

This examination of the *linker* recurrent conversational thread, then, supports our key informant's analysis of the common factors behind such sequential use of threads at LC. It also reveals two additional challenges to conversational coherence posed by this threading activity. One was clearly the much longer temporal span of such recurrent conversational threads—*linker* continued well into LC's second year—requiring that coherence be maintained for much longer. The second additional challenge was uncertainty. The central issues of the five recurrent conversational threads involved difficult and large pieces of the whole LC system. They could not be settled in advance of coding, but had to be kept open and then revisited as the system developed.

We identified two techniques for addressing these additional challenges to maintaining conversational coherence among loosely connected sequential threads. The first was **long-term textual persistence**. Above, we noted the value of short- to mid-term textual persistence for conversational coherence within threads. In these recurrent conversational threads, the temporal span is much longer. Here, long-term storage of email messages for reference, not just short-term storage of messages in the in-box, was a valuable technique. As Dan told us in an interview,

I assumed Robert was archiving.... It was known that I kept my email.

In securing our email data, we learned that Dan did, indeed, keep all of his email, and Robert kept almost all of his. Keith, on the other hand, was the only one of the three full-time LC members who did not retain email regularly. He told us in his interview that he consulted one of

the other two when he needed to refer to a previous exchange. The ability to retrieve previous email exchanges helped LC members reconstruct what had happened at earlier stages of the work, helping to provide some coherence to the loosely linked threads making up a recurrent conversational thread.

The second technique for maintaining conversational coherence in recurrent conversational threads what Girard and Stark (2002) have **provisional settlements**, temporary agreements that enable work to continue in the absence of final agreement. They found that such settlements allowed members of multidisciplinary project teams to (temporarily) reconcile uncertain, incomplete, or incommensurable requirements in order to continue moving forward while acknowledging that further work on the area remains to be done. Even though LC members had relatively similar backgrounds, they also used such provisional settlements to organize their work and, at the same time, to help provide conversational coherence to the loosely connected sequences of threads that comprised recurrent conversational threads. In the last message of the first *linker* thread, for example, Robert referred directly to the provisional nature of their work on the *linker* at this time:

I have hopes that this is the last linker spec for a while. Changes are pretty minor, keith is getting tired of arguing.

An examination of thread closings revealed that in some cases these threads were settled in a manner that was explicitly provisional — that is, the LC members knew that the conversational and work closure was only temporary, and that more remained to be done that would take the cooperation of multiple LC members. In others, the current problem had been addressed, though LC members clearly shared an expectation that related problems would emerge in the future. Thus we classified the settlements to threads making up the recurrent conversational threads by these two general types (see Table 4 for frequencies), finding that over half (51%) were explicitly provisional settlements, with LC members clearly intending to return

to the still open issue later. A significant but smaller set of threads (41%) ended when the immediate problem or issue was solved. Here, the closing did not create an obvious forward-pointing link to the next thread; nevertheless, these threads as a whole also reflected an implicit understanding that the larger issue would continue to be discussed. The remaining threads (8%) were not settled in the electronic conversations, and according to our informant were most likely resolved by discussion in another medium.

Provisional settlements were thus both a particularly salient work practice and an important technique to aide conversational coherence across time and among threads in LC's complex, multi-year product development effort. The ability to temporarily halt discussion on some issues to get on with other business in the expectation of returning to these issues later afforded LC considerable flexibility in when, how, and in what sequence their work was done. It also required, as our key informant noted, mutual commitment to these expectations of recurrence: "you have to trust the people you work with."

4. Conclusions and Implications

Geographically distributed or virtual task teams face many challenges to conversational coherence—challenges that successful task teams must overcome. Our study of the LC electronic mail archive reinforces the importance of conversational threads in asynchronous electronic communication in such distributed task teams. While prior studies of electronic communication have identified the tendency for interactions to coalesce around conversational threads, the specific nature of threading activity and its role in achieving conversational coherence has only rarely been highlighted (e.g., Herring, 1999), and typically not among situated task teams in field settings.

We examined conversational coherence *within, across, and among* threads, identifying three different ways in which LC members used threading to accomplish their distributed, collaborative work and communicative interactions over time: (i) *individual conversational threads*, to carry on tightly linked conversations about a specific issue; (ii) *concurrent conversational threads*, to conduct concurrent but distinct conversations on different topics during the same time period; and (iii) *recurrent conversational threads*, to engage in sequences of conversations that loosely connected related work over longer periods of time. Further, in conjunction with each type of threading activity we studied both the challenges to conversational coherence and the techniques that LC members used to facilitate conversational coherence.

Individual conversational threads, tightly coupled sequences of messages, focus participants' attention and action on a particular issue or topic, providing coherence to a conversation that occurs among distributed members and that takes place over time. LC members faced challenges to conversational coherence in individual conversational threads, including time span and disrupted turn adjacency, problems exacerbated by differences in time zones and work schedules as well as use of dial-up modems and phone lines. To coordinate their work through email, LC members needed to overcome these challenges.

A series of techniques were used to address these challenges to coherence and to highlight the interdependence among the messages making up a thread. Some of these techniques resembled those used in oral conversations to maintain coherence (repeating key lexical terms to reinforce connections, and using summaries and closings in longer threads to remind LC members what the conversation was about and what it had achieved). Others were developed within the context of text-based asynchronous communication and drew on specific features of email (keeping separate subjects in separate messages and threads; using repeated subject lines to indicate connections between messages and migrating subject lines to signal coherence and

progress on the subject of the thread; embedding all or part of a previous message in a response to that message; and taking advantage of short-term textual persistence in email to refer to previous messages). All of these techniques helped establish and maintain coherence within threads, facilitating coordination of their activities around specific issues.

Concurrent conversational threads, parallel threads whose constituent messages are interleaved over time, give participants the opportunity to raise and work on multiple different issues and topics at the same time without sacrificing the coherence of each conversation. Other researchers (McDaniel et al., 1996; Black et al., 1983) have noted that concurrency increases when team members interact via asynchronous (rather than synchronous) electronic tools, since the delay between contributions (built into such tools and exercised by the participants) provides time for additional conversations, a result reinforced by our finding that 88.5% of the threads in the LC email archive were concurrent with at least one other thread. Increased concurrency may decrease the amount of elapsed time required to complete a set of tasks, and it may allow members to make connections across different conversations. We might speculate that such connections could be both valuable (e.g., when serendipitous coupling produces a more integrative view of an issue) and distracting (e.g., when concurrent examination of multiple issues causes confusion or unnecessarily increases complexity).

The practice of concurrent conversational threading creates two additional challenges to conversational coherence: the increased number of conversations going on during the same general time period, and the increased possibility at any given time of a very complex conversational thread.. We identified two additional techniques used by LC members to manage these challenges: thread convergence and divergence. Convergence of initially separate threads when they were found to address the same problem or subject helped LC members reduce the number of threads. Conversely, in some cases a conversational thread that an LC member started to address a problem

such as a system crash was ultimately discovered to be quite complex, involving more than one bug, problem, or issue. The prevalence of concurrent conversational threads suggested a technique for reducing the complexity: unraveling the initial thread into multiple concurrent conversational threads each dealing with a simpler problem. Individuals certainly use both of these techniques in non-mediated oral conversations, but they are particularly useful in an environment that depends heavily on concurrent asynchronous conversations.

Recurrent conversational threads are a phenomenon we have not seen covered elsewhere in studies of electronic communication in on-going, geographically dispersed task teams, but we suspect that it is widespread. We use the term to refer to loosely linked sequences of individual conversational threads. LC members used this threading activity to allocate recurrent attention to key technical topics that were ambiguous, indeterminate, and evolving, thus requiring episodic and emergent discussion. Recurrent conversational threads were used by LC members to coordinate their interactions and maintain continuity in their work over time and across locales. They reflected both the complicated technical issues LC members had to deal with in building their system and the ongoing necessity of coordinating the work each performed independently.

Recurrent conversational threads provided additional challenges to conversational coherence around their much longer time span and around a high level of uncertainty requiring these issues to be kept open for long periods of time. LC members used two techniques to respond to these challenges: long-term textual persistence and provisional settlements. By archiving all or almost all messages, two of the LC members guaranteed that they and the other members could go back to revisit earlier threads within a recurrent conversational thread as necessary to assure coherence. LC members also used provisional settlements (similar to those described by Girard and Stark, 2002) at the end of many individual conversational threads within the sequence of threads making up a recurrent conversational thread. In these cases they made

explicit the fact that the agreement was a temporary expedient to allow them to continue working on other issues, and that it would be revisited at later points in the development of the LC system, thus helping maintain conversational coherence among recurrent conversational threads. Analysis of these recurrent conversational threads suggests the importance—in the ongoing work of any project—of being able to halt a particular interaction for a period of time, settling on some stabilized-for-now agreement (Schryer, 1993), thus making possible the pursuit of other work-related interactions.

The exploratory research reported here has illuminated threading activities and other techniques through which members of a distributed software development team and organization achieved coherence in their email conversations. Our study is primarily based on messages from the first year of LC's existence. As the system they were building became more stable and as the business side of LC became more central to the group, probably fewer conversational threads focused on technical aspects of the system and more on business aspects. Although it is possible that further techniques emerged later in the team's life, it seems likely that the ones identified here continued to be useful.

LC members used a relatively simple medium — electronic mail — as an effective collaborative tool, using a range of threading activities and techniques to enhance conversational coherence within it. A number of researchers (e.g., Majchrzak et al., 2000a; Lewis and Knowles, 1997; Donath et al., 1999; Severinson-Eklundh and Macdonald, 1994) have highlighted the importance of powerful technologies to facilitate collaborative work in distributed teams and online communities. In the case of LC, however, simple electronic mail was used to accomplish a variety of communicative functions, in part aided by a series of techniques—some purely human (e.g., lexical repetition) and others electronically supported (e.g., repeated subject lines)—for maintaining conversational coherence. LC members thus improvised a collaborative

technology using what they had to hand — a simple, low-cost, and relatively robust tool. It suggests that teams can create rich forms of interactions even in the absence of dedicated collaborative technologies.

Additionally, LC members' use of recurrent conversational threads can be seen as a way of helping them manage their ongoing and emerging knowledge. By structuring interactions and decisions so that current deliberations would both reflect on past decisions and lay the basis for future decisions, LC could short-circuit, to some degree, the time needed to develop mutual expectations. As the design was not cast in stone and there was an expectation that key issues would be revisited, mutual understanding was allowed to develop hand-in-hand with the evolving design. This form of engagement resembles the ongoing disputation characteristic of Girard and Stark's (2002) new media projects, and relies on similar provisional settlements.

Further research is needed to examine conversational threading activity in other distributed settings. We believe that our analytic focus on thread use in isolation as individual conversational threads, in parallel in concurrent conversational threads, and sequentially in recurrent conversational threads will be useful in other such settings. We suspect, however, that the extent to which participants in a virtual team use each of these threading activities will vary—both in comparison to LC and within a given team over time—depending on factors such as the nature of the work, the level of interdependence among team members' tasks, changes in focal tasks over time, and the work and personal situations of the members. The repertoire of specific techniques used to achieve conversational coherence within these types of threading activity will likewise vary. Thus we suggest that a particular distributed team and its work may be usefully characterized by its pattern of threading activity and the coherence techniques it uses. This focus on conversational coherence, rather than on individual interactions, should help us better understand how distributed teams enact their virtual work across time and space. We

believe it is only by understanding these dynamic relationships that we can come to understand how the conversational threads created in electronic communications weave “the fabric of activity” (Reder and Schwab, 1990, p.11) that constitutes virtual organizing.

5. Acknowledgements

We would like to thank the members of LittleCompany who generously provided the data for this study. This research was funded by the National Science Foundation under grant number IIS-0085725.

¹ The name of the company, its members, and products have been disguised for confidentiality reasons.

² Indeed, Robert did not meet Martin until well after the period covered by our study, and to this day Dan has not met Martin.

³ Non-threaded messages were typically messages to the group that didn’t require a response (e.g., announcing that a new version of the LC system had been put on the server).

⁴ Twenty percent of the messages (but not of the threads) in our initial archive involved only two parties. Our key informant felt that this proportion was probably typical for the three full-time members. As most of the email in LC originated from the three full-time members, we are missing primarily those exchanges that may have taken place between the other two full-time members.

⁵ According to our key informant, there was a sixth recurrent issue in the first year; however, the LC members rarely referred to this issue in a consistent fashion in subject lines or in the text. Thus, we were unable to distinguish a recurrent conversation around it with the methods we had used to identify the others.

⁶ The thread combining that we did in the recurrent conversational threads subset was not redone in the archive of all threads. We were unable to do such close iterative reading on the entire archive and chose to keep the data in that archive internally consistent.

⁷ These “missing” messages were used only in the analysis of recurrent conversational threads and not in the data on all threads, so as to keep that data on a consistent basis.

⁸ Our distinction between using threads individually and in sequential recurrent conversations bears some resemblance to Reder and Schwab's (1990) distinction between simple and compound episodes of interaction.

⁹ The statistics on thread length Lanzara and Morner (2004) give for a sample of messages in the LINUX and APACHE open source communities (large and amorphous communities) are surprisingly similar to those of the small and well defined LC community. Eliminating the one-message threads (unanswered initial posts which we do not consider threads here) from their data and re-normalizing yields the follow thread-length distribution: 34% and 36% two-message threads in the two communities (compared to 38% in LC) and 22% and 17% three-message threads (compared to 22% in LC).

¹⁰ At the extreme, one exchange that involved waiting for a response from a person outside of LC lasted for 100 days, but this thread is clearly an outlier. The average time span for LC threads is much shorter than those Lanzara and Morner (2004) computed for two open source communities, 2.9 and 1.4 days, respectively, presumably including one-message threads as 0 days and thus actually averaging much longer periods when those single-message "threads" are removed.

¹¹ In this paper we focus on email messages only, though elsewhere we are examining how LC participants combined multiple media in their work.

¹² Thread divergence and convergence are difficult to discern solely by following subject lines. Consequently, although we identified multiple instances of both during close reading of the recurrent threads, we do not have numbers on the cases of divergences and convergence in the entire set of threads.

¹³ Deciding where to divide up the thread and which of the concurrent threads to label as the continuation and which as new threads is a vexing analytic problem, as is deciding when two or more threads converge. Our thread numbers no doubt miss some of both types of cases.

¹⁴ Interestingly, we found that a few threads belonged to, and connected, more than one recurrent conversational thread. Although we have observed that LC members generally communicated about different subjects in separate but often concurrent, threads, the five recurrent conversational threads we identified centered around key technical aspects of the system that interacted with each other. We classified five specific threads as part of two or more different recurrent conversational threads. For example, Thread A in the example above, was classified as belonging to three of the recurrent conversational threads, since the triggering event, a system crash, involved multiple aspects

of the system. This phenomenon underscores the complexity of conversational dynamics among threads in LC's email communication.

¹⁵ These figures are based on the year we studied; they understate the total number of threads and messages because some recurrent conversational threads extended into subsequent years.

6. References

- Baron, Naomi (2004): See You Online: Gender Issues in College Student Use of Instant Messaging. *Journal of Language and Social Psychology*, vol. 23, pp. 397-423.
- Baym, Nancy K. (1996): Agreements and Disagreements in a Computer-Mediated Discussion. *Research on Language and Social Interaction*, vol. 29, pp. 315-346.
- Black, Steven D., James A. Levin, Hugh Mehan, and Clark N. Quinn (1983): Real and Non-Real Time Interaction: Unraveling Multiple Threads of Discourse. *Discourse Processes*, vol. 6, pp. 59-75.
- Chang, Artemis (2005): Synchronicity Matters: Development of Task and Social Cohesion in FtF and Text Based CMC Groups. In *Academy of Management Best Papers Proceedings*, Honolulu, Hawaii, August 5-10, 2005, OCIS Division. Briarcliff Manor, NY: Academy of Management, pp. D1-D6.
- Condon, Sherri L. and Claude G. Cech (1996): Discourse Management Strategies in Face-to-Face and Computer-Mediated Decision Making Interactions. *Electronic Journal of Communication/La Revue Electronique de Communication*, vol. 6, no. 3, (online: http://www.cios.org/getfile/Condon_V6N396).
- Cornelius, Caroline and Margarete Boos (2003): Enhancing Mutual Understanding in Synchronous Computer-Mediated Communication by Training: Trade-Offs in Judgmental Tasks. *Communication Research*, vol. 30, no. 2, pp. 147-177.
- Cramton, Catherine D. (2001): The Mutual Knowledge Problem and Its Consequences for Dispersed Collaboration. *Organization Science*, vol. 12, no. 3, pp. 346-371.

- Donath, Judith S., Karrie Karahalios, and Fernanda B. Viegas (1999): Visualizing Conversation.” In Proceedings of the 32nd Annual Hawaii International Conference on System Sciences, Maui, HI, January 5-8, 1999. Piscataway, NJ: IEEE, p. 2023.
- Ferrara, Kathleen, Hans Brunner, and Greg Whitemore (1991): Interactive Written Discourse as an Emergent Register. *Written Communication*, vol. 8, no. 1, pp. 8-34.
- Galegher, Jolene, Lee Sproull, & Sara Kiesler (1998): Legitimacy, Authority, and Community in Electronic Support Groups. *Written Communication*, vol. 15, no. 4, pp. 493–530.
- Garcia, Angela C. and Jennifer B. Jacobs (1999): The Eyes of the Beholder: Understanding the Turn-Taking System in Quasi-Synchronous Computer-Mediated Communication. *Research on Language and Social Interaction*, vol. 32, no. 4, pp. 337-367.
- Girard, Monique and David Stark (2002): Distributing Intelligence and Organizing Diversity in New Media Projects. *Environment and Planning A*. vol. 34, no. 11, pp. 1927-1949.
- Herring, Susan C. (2004): Computer-Mediated Discourse Analysis: An Approach to Researching Online Behavior. In S. A. Barab, R. Kling, and J. H. Gray (eds.): *Designing for Virtual Communities in the Service of Learning*. New York: Cambridge University Press, pp. 338-376.
- Herring, Susan C. (1999): Interactional Coherence in CMC. *Journal of Computer Mediated Communication*, vol. 4, no. 4 (online: <http://www.ascusc.org/jcmc/vol4/issue4/herring.html>).
- Hiltz, Starr R. and Murray Turoff (1978): *The Network Nation: Human Communication via Computer*. Norwood, NJ: Ablex.
- Hinds, Pamela J., and Mark Mortensen (2005): *Understanding Conflict in Geographically*

Distributed Teams: The Moderating Effects of Shared Identity, Shared Context, and Spontaneous Communication. *Organization Science*, vol. 16, no. 3, pp. 290-307.

Hutchby, Ian (2001): *Conversation and Technology: From the Telephone to the Internet*. Cambridge, U.K.: Polity Press.

Im, Hyun, JoAnne Yates, and Wanda J. Orlikowski (2005): Temporal Coordination through Genres and Genre Systems. *Information, Technology and People*, vol. 18, no. 2, pp. 89–119.

Jones, Quentin (1997): Virtual-Communities, Virtual Settlements & Cyber-Archaeology: A Theoretical Outline. *Journal of Computer-Mediated Communication*, vol. 3, no. 3 (online: <http://jcmc.indiana.edu/vol3/issue3/jones.html>).

Jones, Quentin, Gilad Ravid and Sheizaf Rafaeli (2001): Information Overload and Virtual Public Discourse Boundaries. In: M. Hirose, (ed.), *INTERACT*, Eighth IFIP TC.13 Conference on Human-Computer Interaction, Tokyo, Japan, July 9-13, 2001. Tokyo, Japan: IOS Press.

Kiesler, Sara, Jane Siegel, Timothy W. McGuire (1984): Social Psychological Aspects of Computer-Mediated Interaction. *American Psychologist*, vol. 39, pp. 1123-34.

Lanzara, Giovan F. and Michele Morner (2004): Making and Sharing Knowledge at Electronic Crossroads: The Evolutionary Ecology of Open Source. Paper presented at the Fifth European Conference on Organizational Knowledge, Learning and Capabilities, Innsbruck, Austria, April 2-3, 2004.

Lewis, David D. and Kimberly A. Knowles (1997): Threading Electronic Mail: A Preliminary Study. *Information Processing and Management*, vol. 33, no. 2, pp. 209-217.

- Lipnack, Jessica and Jeffrey Stamps (1997): *Virtual Teams: Reaching Across Space, Time, and Organizations with Technology*. New York: John Wiley & Sons.
- Majchrzak, Ann, Ronald E. Rice, Arvind Malhotra, Nelson King, and Sulin Ba (2000a):
Computer-mediated Inter-organizational Knowledge-Sharing: Insights from a Virtual Team using a Collaborative Tool. *Information Resources Management Journal*, vol. 13, no. 1, pp. 44-54.
- Majchrzak, Ann, Ronald E. Rice, Arvind Malhotra, Nelson King, and Sulin Ba (2000b):
Technology Adaptation: The Case of a Computer-Supported Inter-organizational Virtual Team. *MIS Quarterly*, vol. 24, no. 4, pp. 569-601.
- Markus. M. Lynne (1994): Electronic Mail as the Medium of Managerial Choice. *Organization Science*, vol. 5, no. 4, pp. 502-527.
- Markus, M. Lynne, Brooke Manville, and Carole E. Agres (2000): What Makes a Virtual Organization Work? *Sloan Management Review*, vol. 42, no. 1, pp. 13-26.
- Maznevski, Martha L. and Katherine M. Chudoba (2000): Bridging Space Over Time: Global Virtual Team Dynamics and Effectiveness. *Organization Science* vol. 11, no. 5, pp. 473-492.
- McDaniel, Susan E., Gary M. Olson, and Joseph C. Magee (1996): Identifying and Analyzing Multiple Threads in Computer-Mediated and Face-to-Face Conversations. In *Proceedings of the 1996 ACM Conference on Computer Supported Cooperative Work*, Boston, MA, November 16-20, 1996. ACM Press, Cambridge, MA, pp. 39-47.
- Millen, David R., and John F. Patterson (2002): Stimulating Social Engagement in a Community Network. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, New Orleans, LA, November 16-20, 2002. New York: ACM Press, pp. 306-313.

- Murray, Denise E. (1985): *Composition as Conversation: The Computer Terminal as Medium of Communication*. In L. Odell and D. Goswami (eds.): *Writing in Nonacademic Settings*. New York: The Guilford Press, pp. 203-228.
- Murray, Denise E. (1988): *The Context of Oral and Written Language: A Framework for Mode and Medium Switching*. *Language in Society*, vol. 17, no. 3, pp. 351-373.
- Murray, Denise E. (2000): *Protean Communication: The Language of Computer-Mediated Communication*. *TESOL Quarterly*, vol. 34, no. 3, pp. 397-421.
- Nardi, Bonnie A., Steve Whittaker, and Erin Bradner (2000): *Interaction and Outeraction: Instant Messaging in Action*. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*, Philadelphia, PA, December 2-6, 2000. New York: ACM Press, pp. 79-88.
- Orlikowski, Wanda J., and JoAnne Yates (1994): *Genre Repertoire: The Structuring of Communicative Practices in Organizations*. *Administrative Science Quarterly*, vol. 39, no. 4, pp. 541-574.
- Reder, Stephen, and Robert G. Schwab (1990): *The Temporal Structure of Cooperative Activity*. In *Proceedings of the 1990 ACM Conference on Computer Supported Cooperative Work*, Los Angeles, CA, October 7-10, 1990. New York: ACM Press, pp. 303-316.
- Reed, Darren (2001): 'Making Conversation': Sequential Integrity and the Local Management of Interaction on Internet Newsgroups." In *Proceedings of the 34th Hawaii International Conference on System Sciences*, Maui, HI, January 3-6, 2001. Piscataway, NJ: IEEE, p. 4035.
- Reinsch, N. Lamar, Jr., Jeanine W. Turner, and Catherine H. Tinsley (2006): *Five Conversations at Once: Multi-Communicating in the Workplace*. Working paper, McDonough School of

Business, Georgetown University, Washington, D.C.

Rennecker, Julie A. (2001): *The Myth of Spontaneous Connection: An Ethnographic Study of the Situated Nature of Virtual Teamwork*. Ph.D. dissertation, Sloan School of Management, Massachusetts Institute of Technology, MA.

Rheingold, Howard (1993): *The Virtual Community: Homesteading on the Electronic Frontier*. Reading, MA: Addison-Wesley Publishing Company.

Schryer, Catherine F. (1993): Records as Genres. *Written Communication*, vol. 10, no. 2, pp. 200-234.

Schmidt, Kjeld and Ina Wagner (2004): Ordering Systems: Coordinative Practices and Artifacts in Architectural Design and Planning. *Computer Supported Cooperative Work*, vol. 13, nos. 5-6, pp. 349-408.

Severinson-Eklundh, Kerstin and Macdonald, Clare (1994): The Use of Quoting to Preserve Context in Electronic Mail Dialogues. *IEEE Transactions On Professional Communication*, vol. 37, no. 4, pp. 197-202.

Sproull, Lee and Sara Kiesler (1986): Reducing Social Context Cues: Electronic Mail in Organizational Communication. *Management Science*, vol. 32, no. 11, pp. 1492-1512.

Walther, Joseph B. (2002): Time Effects in Computer-Mediated Groups: Past, Present, and Future. In P. Hinds and S. Kiesler (eds.): *Distributed Work*. Cambridge, MA: MIT Press, pp. 235-257.

Whittaker, Steve, Quentin Jones, and Loren Terveen (2002): Managing Long Term Communications: Conversation and Contact Management. In *Proceedings of the 2002*

Conference on Computer Supported Cooperative Work, New Orleans, LA, November 16-20, 2002. New York: ACM Press, pp. 216-225.

Yates, JoAnne and Wanda J. Orlikowski (1993): Knee-Jerk Anti-LOOPism and Other E-Mail Phenomena: Oral, Written & Electronic Patterns in Computer-Mediated Communication. Sloan working paper #3528-93, Sloan School of Management, Massachusetts Institute of Technology, MA.

Yates, JoAnne, Wanda J. Orlikowski, and Kazuo Okamura (1999): Explicit and Implicit Structuring of Genres in Electronic Communication: Reinforcement and Change of Social Interaction. *Organization Science*, vol. 10, no. 1, pp. 83-103.

Yates, JoAnne, Wanda J. Orlikowski, and Julie A. Rennecker (1997): Collaborative Genres for Collaboration: Genre Systems in Digital Media. In *Proceedings of the 30th Annual Hawaii International Conference on System Sciences*, Maui, HI, January 7-10, 1997. Piscataway, NJ: IEEE, pp. 50-59.

Table 1. Threading Activity in the LC Email Archive: February 1997 – February 1998

Number of threads	469
Number of messages	2,163
Average number of messages/thread	4.6
Median number of messages/thread	3
Shortest thread	50 seconds
Longest thread	74 days, 23 hours, 37 minutes, 45 seconds
Longest thread including only LC members	23 days, 18 seconds
Median elapsed time/thread	7 hours, 29 minutes, 1 second
Average elapsed time/thread	27 hours, 13 minutes, 28 seconds
<hr/>	
Number of 2-interaction threads	179
Number of 3-interaction threads	103
Number of messages in longest thread	32

Table 2. LC Threading Concurrency

% of threads with no overlap	11.5%
% of threads concurrently overlapping a maximum of 1 thread	30.2%
% of threads concurrently overlapping a maximum of 2 threads	33.6%
% of threads concurrently overlapping a maximum of 3 threads	17.2%
% of threads concurrently overlapping a maximum of 4 threads	4.7%
% of threads concurrently overlapping more than 4 threads	2.8%
% of threads concurrent with at least one thread	88.5%
Median number of threads concurrent at one time	3
Largest number of threads concurrent at one time	7

Table 3. Summary of LC Recurrent Conversational Threads

Recurrent Conversational Thread	Initial Number of Threads	Total Threads After Combining	Total Number of Messages	Median Number of Messages per Thread	Average Number of Messages per Thread
Linker	35	35	280	5	8.00
Hash	5	5	25	5	5.00
GC	20	19	140	5	7.37
BigNums	8	7	36	3	5.14
Make	29	23	188	5	8.17
Total	97	89			

Table 4. Settlement Types in LC Recurrent Conversational Threads

Recurrent Conversational Threads	<i>Thread Settlements</i>		
	<i>Provisional</i>	<i>Apparently Settled</i>	<i>Unsettled</i>
Linker	65%	24%	11%
Hash	60%	40%	–
GC	37%	63%	–
BigNums	43%	43%	14%
Make	43%	48%	9%
Across all five conversations	51%	41%	8%

Figure 1. Distribution of Simple Threads in LC Archive

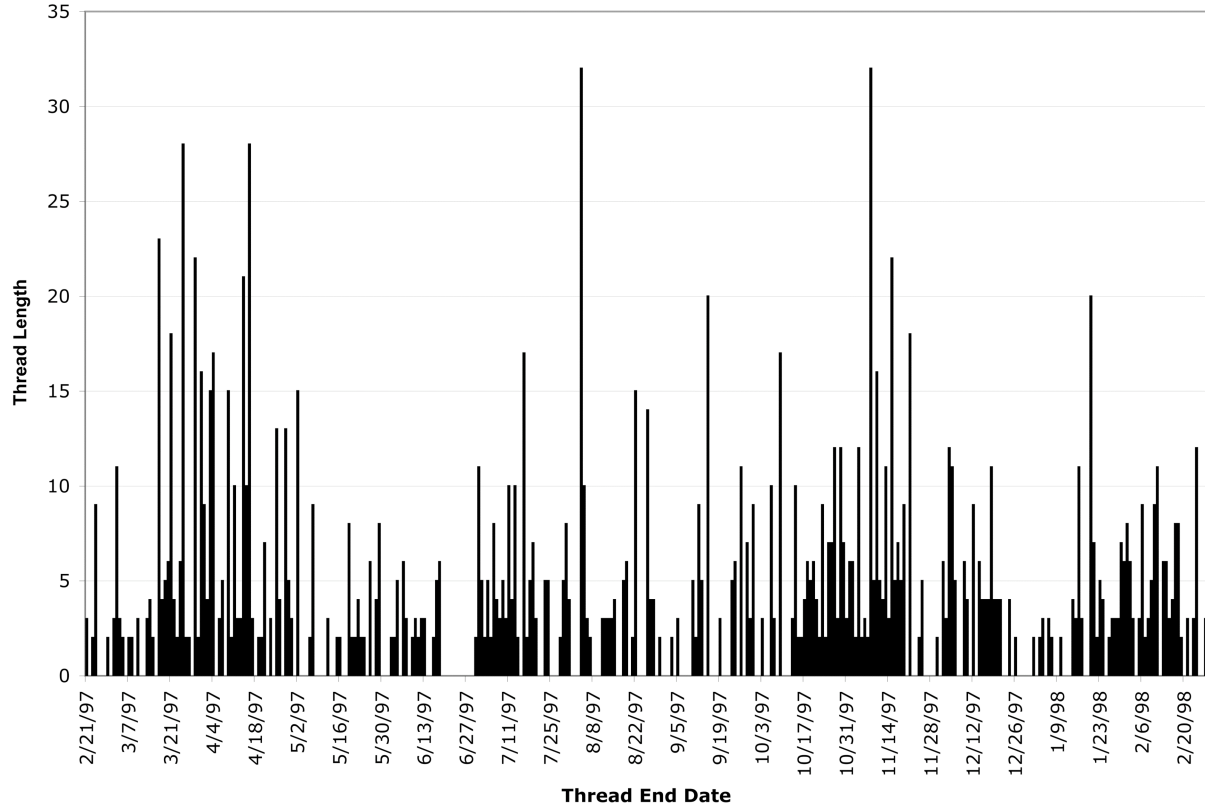


Figure 2. Thread Concurrency (Times in GMT)

Message initiated by:

■ Martin

□ Dan

■ Keith

■ Robert

Number of threads addressing all of LC = 4
 Number of messages = 56

Thread A = 32 messages
 Thread B = 16 messages
 Thread C = 6 messages
 Thread D = 2 messages

Side threads between two LC members = 1
 Thread E = 3 messages

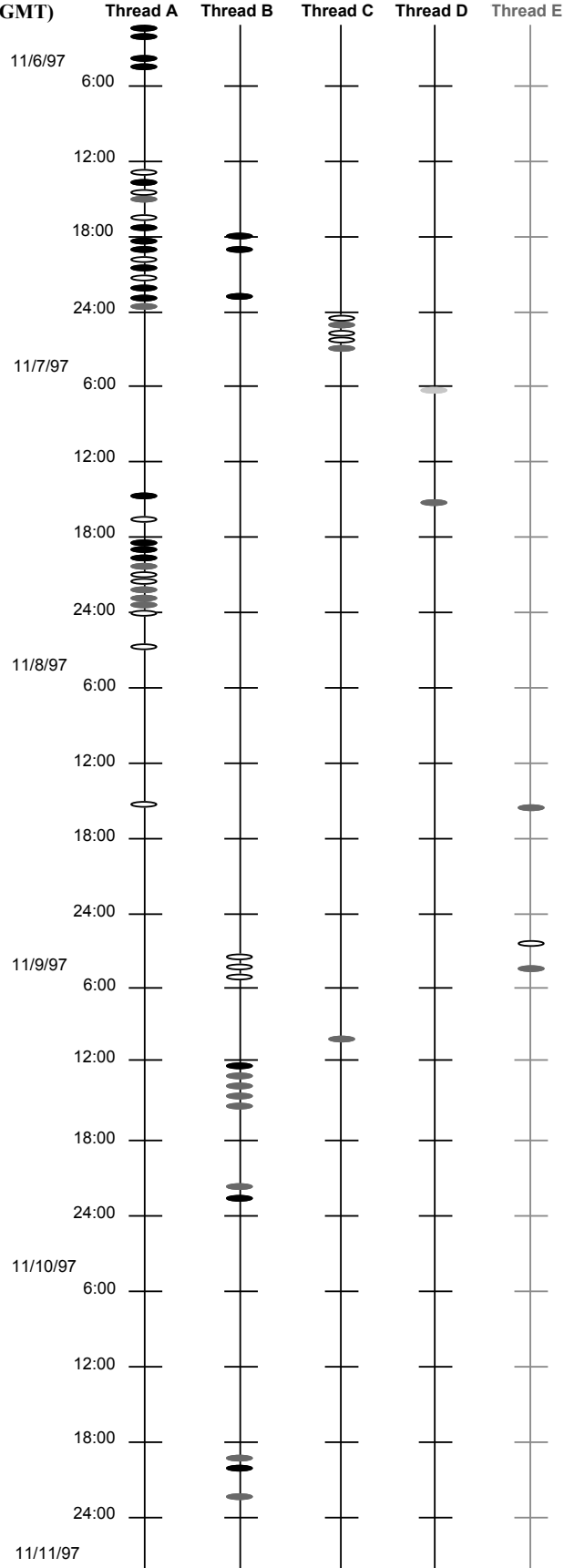


Figure 4. Thread divergence to reduce complexity (simplified diagram)

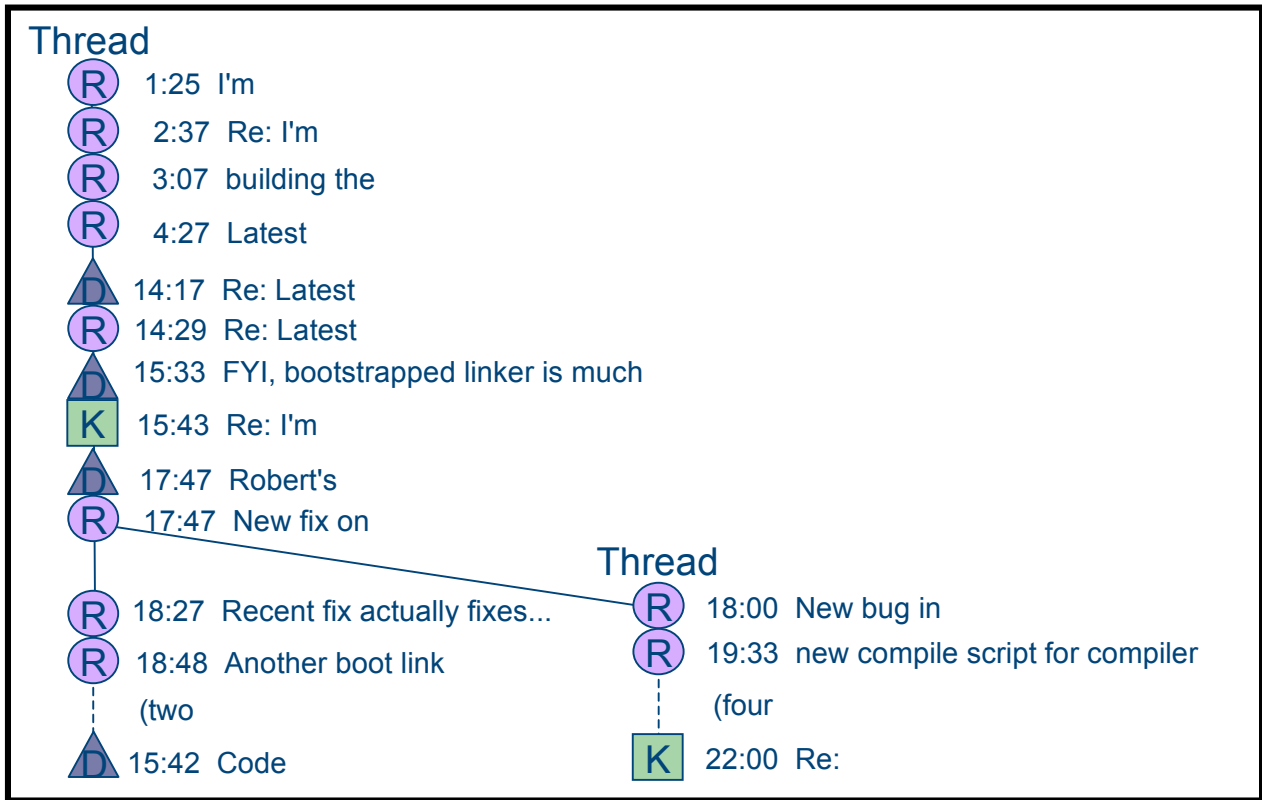


Figure 5. Distribution of Threads in *Linker* Recurrent Conversational Thread

