Constructing collaborative ecologies: how selection, practice, and mediation assemble and shape social and collaborative software

David Niedergeses

Iowa State University

Follow this and additional works at: http://lib.dr.iastate.edu/etd

Part of the Rhetoric Commons

Recommended Citation
Niedergeses, David, "Constructing collaborative ecologies: how selection, practice, and mediation assemble and shape social and collaborative software" (2012). Graduate Theses and Dissertations. 12559.
http://lib.dr.iastate.edu/etd/12559

This Dissertation is brought to you for free and open access by the Graduate College at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Constructing collaborative ecologies: how selection, practice, and mediation assemble and shape social and collaborative software

by

David Matthew Sertich Niedergeses

A dissertation submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Co-majors: Rhetoric and Professional Communication; Human Computer Interaction

Program of Study Committee:
Gregory D. Wilson, Co-major Professor
Anthony M. Townsend, Co-major Professor
Geoffrey F. K. Sauer
Brian E. Mennecke
Katherine Gossett
Richard Crosby

Iowa State University
Ames, Iowa
2012

Copyright © David Matthew Sertich Niedergeses, 2012. All rights reserved.
For Jordan,

my loving companion from the beginning
# Table of Contents

List of Tables .................................................................................................................. v  
List of Figures ................................................................................................................... vi  
Acknowledgements ......................................................................................................... vii  
Abstract .......................................................................................................................... viii  

Chapter 1: The Construction of Collaborative Ecologies ................................................. 1  
  Mapping a User Experience Team’s Collaborative Ecology ........................................ 2  
    Description of the Study ............................................................................................. 3  
    Threads of User Experience Activity ......................................................................... 4  
    User Experience Tool Ecology .................................................................................. 6  

Ecology Metaphors ......................................................................................................... 7  
  Software Ecosystems .................................................................................................. 9  
  Information Ecologies ................................................................................................ 10  
  Media and Genre Ecologies ......................................................................................... 12  
  Implications of Ecology Metaphors ............................................................................ 15  

Assembling and Shaping Collaborative Ecologies ......................................................... 16  
  Selection ..................................................................................................................... 18  
  Practice ....................................................................................................................... 19  
  Mediation ................................................................................................................... 20  
  Theoretical Alignments of Selection, Practice, and Mediation ................................ 21  

Conclusion ...................................................................................................................... 22  

Chapter 2: Theorizing a Rhetoric for Collaborative Ecologies ....................................... 23  
  Artifact Usability ....................................................................................................... 23  
    Usability of Computers ............................................................................................ 23  
    Usability of Communication .................................................................................... 25  
  Sociocultural Usability ............................................................................................... 27  
    Activity Theory Defined ......................................................................................... 29  
    Activity Theory in Workplace Studies .................................................................... 32  
    Activity Theory and Genre Theory ......................................................................... 36  
    Actor-Network Theory Defined .............................................................................. 39  
    Actor-Network Theory in Workplace Studies ......................................................... 41  

Rhetoric for Collaborative Ecologies ............................................................................. 44  
  Selection ..................................................................................................................... 45  
  Practice ....................................................................................................................... 47  
  Mediation ................................................................................................................... 49  

Conclusion ...................................................................................................................... 50
Chapter 3: Methods

Methodological Choices
- Choice 1: Ethnography for Human Centered Design
- Choice 2: The Actor-Network Theory Understanding of Group Formation
- Choice 3: Viewing Collaborative Ecologies as an Assemblage
- Choice 4: Qualitative Interviewing as the Primary Means of Data Collection

Procedural Methods
- Research Site Requirements and Description
- Relationship to Participants
- Data Collection
- Data Analysis

Conclusions

Chapter 4: A User Experience Collaborative Ecology

Thread 1: Supporting the Software Development Life Cycle
Thread 2: Collecting User Data
Thread 3: Supporting User Requirements
Thread 4: Supporting High-Level Designs and Specifications
Thread 5: Supporting Design and Graphic Prototypes

Conclusion

Chapter 5: Selection, Practice, and Mediation

Three Mechanisms in Five Threads
- Selection
- Practice
- Mediation

Beyond Selection, Practice, and Mediation
Idiosyncrasies of the Study
Ecology Design

Appendix A: Letter of Informed Consent
Appendix B: Interview Protocol
References
Biographical Sketch
List of Tables

Table 1: The user experience team's prominent activities .................................................. 5
Table 2: Selection, practice, and mediation alignments .......................................................... 21
Table 3: Relationships between selection, practice, mediation, and theory. .......................... 45
Table 4: Summary of study participants .................................................................................. 67
Table 5: Preliminary coding scheme and associated narrative thread ..................................... 70
Table 6: Emerging threads of activity in the user experience collaborative ecology ............. 72
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Formulation of activity theory (From Uden, Valderas, and Pastor 2002)</td>
<td>32</td>
</tr>
<tr>
<td>2</td>
<td>Tools supporting SDLC management</td>
<td>79</td>
</tr>
<tr>
<td>3</td>
<td>Tools supporting data collection and customer relationships</td>
<td>93</td>
</tr>
<tr>
<td>4</td>
<td>Tools supporting the identification and prioritization of requirements</td>
<td>101</td>
</tr>
<tr>
<td>5</td>
<td>Specification documents progress through three stages</td>
<td>103</td>
</tr>
<tr>
<td>6</td>
<td>Supporting high-level designs and specifications</td>
<td>110</td>
</tr>
<tr>
<td>7</td>
<td>Supporting graphic design and prototyping</td>
<td>122</td>
</tr>
</tbody>
</table>
Acknowledgements

The current form of this dissertation is the result of a great deal of time, thought, and effort, and not all of it mine. I owe a great deal to the patient work of Greg Wilson, my adviser in the English Department. Aside from me, he has poured more time, and thought into this project than anyone. I cannot begin to measure his contribution. I must also recognize my advisor in Human Computer Interaction, Tony Townsend, for serving as my guide and advocate in my attempts to bridge disciplines. The remaining members of my program of study committee, Geoff Sauer, Brian Mennecke, Kathie Gossett, and Ben Crosby, also deserve thanks for all of the meetings, comments, critiques, and encouragement that have been vital to my work.

While these six saw me through to the end and have my sincerest appreciation, dozens of others helped me on the journey. They served on committees, instructed me in course upon course, and helped me through procedural hurdles over seven years of graduate study. I cannot list them all, but Lee Honeycutt, Carl Herndl, and Rebecca Burnett would appear at the top.

Finally, I thank my wife Jordan, who has sacrificed more for this project than anyone, perhaps including myself. We may not have known exactly what we were in for when I started this, but it is done now. I hope you are proud.
Abstract

This dissertation examines how a user experience team at a multinational corporation transforms a collection of software applications into a socially usable collaborative ecology. Collaborative ecologies are sociocultural systems that consist of persons, activities, tools, and ideas that are mutually constructive. The metaphor of ecology, which has emerged in the disciplines of human computer interaction, computer supported cooperative work, and rhetoric and professional communication, informs an ethnographic inquiry that includes seven months of daily immersion and ten hours of qualitative interviews. Drawing on a diverse reading of interdisciplinary theory, including traditional usability studies, genre theory, activity theory, and actor-network theory, the dissertation distills the construction of collaborative ecologies into three mechanisms: the selection of tools, the development of practices, and the mediation of ideas about those tools and practices. Applying selection, practice, and mediation in the context of the ethnographic study generates insights about the user experience team’s activities, about the collaborative ecology that support them, and about how selection, practice, and mediation operate. These insights are useful for the design and facilitation of social and collaborative software systems because they suggest a way to understand the role that users, activities, tools, and ideas play in constructing their ecology.
Chapter 1: The Construction of Collaborative Ecologies

A designer selects a free, open source image editor instead of requesting that his manager purchase a license for an industry-standard commercial package. His coworker maintains two, three, or more groupware tools for sharing files even though they seem to do basically the same thing. Another coworker, improvising an analysis tool for one project, does not share that tool with a user feedback specialist in another group who desperately needs it every month.

These are just a few anecdotes that indicate how transforming a loose connection of software tools into an efficient, effective, and satisfying collaborative ecology is a challenging and messy task that may not make sense to outsiders. Yet, logically, it is a challenge that every successful team continually overcomes. Professionals collaborate incessantly, and use dozens of tools to help. Making decisions and building consensus about how to use which tools requires communal effort that results in a collaborative ecology consisting of people and their tools, activities, and ideas. The question this study asks is: how do actual groups of workers assemble and shape their collaborative ecologies?

In this chapter I describe the details and background of that question. First, I outline a study for mapping a user experience team’s collaborative ecology. Second, I review various definitions of the term ecology, which is offered by three disciplines as a metaphorical approach to understanding sociocultural tool use. Finally, I propose three mechanisms—selection, practice, and mediation—that assemble and shape a collaborative ecology. These mechanisms, drawn from a broad reading of interdisciplinary theory, represent one way to understand the narrative that emerged from the research site.
Mapping a User Experience Team’s Collaborative Ecology

I use the phrase “collaborative ecology” to indicate a social use of shared tools that is a contingent, dynamic, mutually-constitutive endeavor where the people and their activities, tools, and ideas are all interdependent. Our collaborative ecologies are getting more complex all the time. We spend a growing portion of our time living and working in cyberspace.

Distributed working environments are common (Haythornwaite, 2004, p. 2). Innovations in social media reshape how we coordinate behavior (Shirky, 2009, pp. 48, 87, 104-108). Internet access has become ubiquitous, continuous, and egalitarian (Morville, 2005, pp. 65-66). Computing paradigms are shifting in favor of “innovations in Web-based technologies” that change “how we are coming to understand computer networks,” (Diehl, Grabill, & Hart-Davidson, 2008, p. 423). The increasing complexity of our collaborative ecologies challenges groups of professional workers.

In this project, I use ethnographic observations and interviews to examine how a software user experience team transforms a collection of software applications into a usable collaborative ecology to support their work. This ethnographic process produced a narrative of five of the group’s most important activities and a map of the collaborative tools they use to support their activities. The narrative and map help to illustrate how selection, practice, and mediation assemble and shape the collaborative ecology.

This section summarizes the methods I used in this ethnographic study which are further explained in chapter 3. First, it describes the basic methods of the study of a user experience team’s collaborative ecology. Second, it identifies five threads of activity that emerged from the ethnographic study. Finally, it describes the user experience team’s collaborative ecology.
Description of the Study

To learn how a team assembles and shapes a collaborative ecology—how they transform a collection of software tools, activities, and ideas into a successfully working system, I studied a user experience team. Drawing on ethnographic observations and qualitative interviews during seven months of embedded inquiry, I discovered social and rhetorical activities that transform collections of software into effective collaborative environments. Along the way, I found myself mapping the team’s collaborative ecology in general, including the activities that make up their daily work and the tools that support those activities.

The user experience team that I studied is a small part of a large multinational corporation. The primary group is a small team of about twenty professionals. The precise number and organizational structure depends on the projects ongoing and on shifts in the broader corporation’s hierarchy. Roughly half of the team is based in a mid-sized city in the American Midwest, while other members work remotely and reside in other cities and countries. Working remotely at least part of the week is common, even for members at the primary work site.

Data on the user experience team’s collaborative ecology was gathered with ethnographic observation and qualitative interviewing. Observation occurred as part of a seven month immersion experience during which I interacted with the team daily. In addition, I conducted interviews with select members of the team which were digitally recorded and then transcribed for analysis. The open-ended character of data collection reflects the naturalistic approach of ethnographic research, and it recognizes the need for a flexible partnership between the participants and the researcher.
Confidentiality is important for collecting data from participants. My research site is a place of business, and has legitimate concerns regarding confidential and proprietary information and over the privacy of their employees. Because of my agreements with the research site, I could not actively take field notes during the immersion phase of the project and instead relied on later reflection. Interviews were subject to official consent of participants on individual and organizational levels, and the consent and data collection processes were reviewed by the Iowa State University Institutional Review Board.

The project used qualitative data analysis. Study data was qualitatively coded based on the activities that emerged as organizing principles in the user experience team. The analysis was actively adapted during coding to address the patterns emerging from the data. I used RQDA (R Project, 2012) for qualitative coding; RQDA is a free and open source qualitative data analysis program that is developed on the R programming language. RQDA supported data analysis by organizing key passages and texts and by allowing the researcher to explore these passages in a variety of ways. The data was then examined for the three mechanisms drawn from theory (selection, practice, and mediation) that assemble and shape a collaborative ecology.

**Threads of User Experience Activity**

As I explain in chapter 4, five threads of activity emerged from the ethnographic case of a user experience team’s collaborative ecology (Table 1 next page). Each activity is tied to the responsibilities of the user experience team I researched, and each is supported by a distinct collection of tools, activities, and ideas.
Table 1: The user experience team’s prominent activities

<table>
<thead>
<tr>
<th>Thread</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The Software Development Life Cycle</td>
</tr>
<tr>
<td>2</td>
<td>Collecting and Reporting User Data</td>
</tr>
<tr>
<td>3</td>
<td>Identifying and Prioritizing Requirements</td>
</tr>
<tr>
<td>4</td>
<td>High-Level Designs and Specifications</td>
</tr>
<tr>
<td>5</td>
<td>Design and Graphic Prototypes</td>
</tr>
</tbody>
</table>

The most prominent activity of the user experience team is their participation in the software development lifecycle, or SDLC. The SDLC is a general concept for managing software projects from inception to completion. It is the overriding cycle that seems to regulate the activity of the entire project team, including user experience, software development, marketing, and testing teams.

The second thread of activity in the user experience collaborative ecology is collecting and reporting user data. The user experience team derives most of its authority within the organization from its relationship to end users. The second thread represents the team’s relationship to those users, including how the team gathers information from them, packages it, and then stores it as a source of authority to use in their other activities.

The third thread has to do with identifying and prioritizing new software requirements. The user experience team shares this activity with many other stakeholders. The team is responsible for injecting user-centered requirements into the design process and then shepherding those requirements through development.

The fourth thread involves writing high-level designs and specifications. Specifications are collaboratively written documents that express the changes and
improvements planned for a new release of a product, or any requirement for a new product. The user experience team has a stake in everything that affects the end users’ perceptions of the project, but they have a particular responsibility for working with the visual, interface, and interaction design.

The last thread is where individual users create design prototypes to communicate designs to developers, managers, customers, and other stakeholders. They circulate designs in order to get feedback that can help refine their work, but also to verify and test designs.

All five threads are all intertwined, with each one depending on the others and often occurring at the same time. It is common for the output of one thread of activities to circulate through other threads since at some level they are all part of the same activity system. Furthermore, the threads relate to the overall software development methodology of the project team. Methodologies like Agile software development have a profound influence on individual and shared activity across entire projects, including the user experience team.

**User Experience Tool Ecology**

The user experience team I studied participates in and helps to shape a collaborative ecology built of over 50 software applications, according to my conservative count. These applications include everything from rigid project management tools that reflect and enforce Agile methodologies to highly flexible and open-ended tools that can fill a variety of roles, like a corporate wiki program. A secondary goal of my case narrative is to map the collaborative ecology to see relationships between tools, activities, and ideas in the organization.
Mapping the collaborative ecology revealed insights about how the distribute tasks across their collection of tools. For example, the team tends to use several different groupware tools for storing files, largely for political reasons about controlling who can access their files and how those readers will perceive it. Another example is the handful of occasions when team members select free and open source software packages instead of commercial ones, thereby avoiding official channels for software procurement. Some of the most interesting events occur when local understanding of tools uses come into tension with each other or with organizational standards. At other times, the opposite happens; participants in one corner of the organization identify needs that have already been addressed in other corners, but the solutions have not mediated back through network.

These are just a couple of the many events and occurrences that emerged from the case narrative. But the account does more than just catalogue the tools. The emerging map is a representation of a collaborative ecology that is assembled and shaped by selecting tools, developing practices, and mediating ideas. In order to understand how these three mechanisms work, we need a deeper understanding of the ecology metaphor for analyzing sociocultural tool use.

**Ecology Metaphors**

The ecology metaphor I use in this study has a history in three related discipline, including human computer interaction (HCI), computer supported cooperative work (CSCW), and rhetoric and professional communication (RPC). Classical usability studies in HCI usually take artifacts of technology as their focus. Studies in CSCW take artifacts of human behavior. Studies in RPC take artifacts of communication. While each of the three
Disciplines has a different initial focus, over the last three decades each has tended to expand from artifact-centric inquiry to sociocultural inquiry. Because of this expansion all three disciplines have come to share a broader region of concern. Though each discipline began with the goal of improving different kinds of artifacts, each one ends up trying to improve the very same network of people, tools, activities, and ideas. And, all of them have developed metaphors based on ecology to describe that network.

The different metaphors applied by researchers from HCI, CSCW, and RPC add nuance to a common understanding that the use of collaborative and social software is a contingent, dynamic, mutually constitutive endeavor. Understanding any part of it—the people, the tool, the activity, or the ideas—requires understanding the whole. The particular metaphors, including information ecology, genre ecology, media ecology, and ecosystem, along with system, assemblage, environment, and context all seek to explain that whole.

These ecology metaphors and the theoretical traditions that have informed them are not the only way to address dynamic networks of people, tools, activities, and ideas. One alternative from Anthony Giddens (1974) is structuration theory. As Herndl explains, Giddens’s structuration shifted analysis from structures to “structural properties” that “only exist in the real time of social activity and have to be constantly maintained by the actions and memory of social agents,” (1996, p. 459). Structuration has been particularly useful for examining technology systems in the work of adaptive structuration theory, developed by DeSanctis and Poole (1994). Another alternative descends from Burke’s dramatist pentad of act, agent, agency, purpose, and scene (1969). As explained by Dayton, the dramatist pentad provides “an initial structure into an investigation into motives, which can then be shifted by analyzing one perspective as mediated by another,” (2006, p. 363). Both adaptive
structuration theory and the dramatist pentad are powerful ways to analyze complex systems of people, activities, tools, and ideas that run parallel to this project. Instead of developing them further, this project examines ecology as a metaphor for sociocultural systems.

To develop a deeper understanding of ecology metaphors in general, the next section reviews some of the most prominent examples. First, I explain software ecosystems, which consider software applications first. Second, I examine information ecologies, which consider technological behavior first. Finally, I examine media and genre ecologies, which consider documents and ideas first.

**Software Ecosystems**

“Software Ecosystem” (SECO) is an ecology metaphor that emphasizes the role of software applications. Jansen, Finkelstein, and Brinkemper define SECOs as “a set of businesses functioning as a unit and interacting with a shared market for software and services, together with the relationships among them,” (2009, ¶6). Draxler and Stevens (2011) use an “open software ecosystem” model to examine the Eclipse integrated development environment (IDE), demonstrating how sociocultural activity enriches our understanding of everything from software development by hobbyists, professionals, and corporations, to local appropriation of practices by experienced versus inexperienced software developers. They recognize a paradigm shift stemming from the Internet’s dominance for mass communication, from growth in digital distribution of products, business and development models exploiting “gift culture”, from increase in users sharing software, and most importantly from “the establishment of loosely coupled networks of manufacturers,
Eclipse arose as the product of “loosely coupled networks of manufacturers, semi-professionals, and hobbyists, creating small-scale components which can be individually assembled by users,” (Draxler & Stevens, 2011, p. 404). Because Eclipse is an open source platform that is highly customizable through the use of software plug-ins, the presumption is that each user can tailor the platform to her own needs (p. 423). According to Draxler and Stevens’ mixed-methods examination, individual users do take advantage of the flexibility of the software. However, even though Eclipse is ostensibly a single-user tool, the drive to customize is balanced against a need for commonality between separate users. They write,

Another key result of this study is to show that designing the workplace by making use of software ecosystems is not a competence of the individual user, but a collective competence of the workgroup or whole company. This collective competence is maintained in various situations like regular team meetings, break downs, asking for help or introducing juniors to a new field of work. (p. 433)

One of the themes in Draxler and Stevens’s study is that developers’ relationship to the Eclipse platform was coordinated with their membership in social groups. Younger developers use Eclipse heavily, with lots of customization, while employees who began work prior to Eclipse’s widespread use prefer the streamlined use of simple text editors as a development platform (p. 427).

**Information Ecologies**

“Information Ecologies” is an ecology metaphor that emphasizes the information technology practices of groups of people. Information ecologies are people-first and activity-first ecologies. In *Information ecology: Using technology with heart*, Nardi and O’Day define “information ecology” as a “system of people, practices, values, and technologies in a
particular local environment,” (Nardi & O’Day, 1999, p. 49). They prefer information ecology over other models because it focuses on the “human activities that are served by technology,” (p. 49). They pass over system, text, and tool modes, noting that all three have virtues and drawbacks. They take issue with the “formidably complex and essentially pessimistic picture” suggested in the “system” metaphors associated with Jacques Ellul (1964), Langdon Winner (1977), and Neil Postman (1993) (p. 27), even while they recognize that “system” offer the “richest, most troubling, and most mind altering perspective.” (33).

Nardi and O’Day also set aside the “text” metaphor of actor-network theorists like Latour (1988) and Callon (1991), arguing that analyzing technology as “text” does little to account for issues like learning and changing behavior, judgment, creativity, values, and other social dynamics (p. 33). Finally, the “tool” metaphor, while it addresses important usability concerns, fails to account for “issues that extend beyond individual humans,” like the social, organizational, and political context that is often important to understanding a technology (p. 30).

In their critique, Nardi and O’Day identified Neil Postman as a proponent of a “system” metaphor, but scholars who follow his work also associate him with a separate media ecology tradition. Susan Barnes is a contemporary example of the media ecology tradition. Her essay in Mediated Interpersonal Communication (2008) gives a media ecology analysis of the current day social media. Media ecology, which is descended from the theory of Walter Ong (1975), Marshall McLuhan (1964), and Neil Postman (1993), describes interpersonal communication as actions within an environment. The nature and possibility of that interpersonal communication depends on the environment. As the environment changes, so does the nature and possibility of interpersonal communication (Barnes. 2008, p. 15).
When read in context with other social theories, Barnes’s explanation of media ecology adds a uniquely humanistic, “interpersonal” perspective to the notion of communication within a network or social structure. It handles characteristically “communicative” elements of networks that others do not emphasize.

**Media and Genre Ecologies**

“Media and genre ecologies” emphasize how ideas and information are handled in an ecology. They are symbol-first ecologies. Scholars in rhetoric and professional communication have developed media and genre ecologies in conjunction with information ecologies. For example, Hart-Davidson, Bernhardt, McLeod, Rife, & Grabill take up Nardi and O’Day’s “information ecology” as the framework for “Coming to content management: Inventing infrastructure for organizational knowledge work,” (2008). In their discussion they identify fourteen considerations for organizations undergoing a major shift in technology like deploying a content management system (p. 11-12). Information ecologies help them examine the outcome of these considerations in their analysis of a library website.

Perhaps the best synthesis of ecology metaphors from a rhetorical perspective comes from Colin Gifford Brooke, who recognizes their usefulness in the work of many scholars. Brooke’s book, *Lingua Fracta*, reviews ecology metaphors that have influenced new media. As he writes,

> Ecologically, practice includes all of the ‘available means’ and our decisions regarding which of them to pursue. In the case of interfaces, this ecology also includes not only those practices involved in the production of a particular interface, but those made possible by it. (p. 49)

He seeks to re-envision the classical rhetorical canons of invention, arrangement, style, memory, and delivery for new media. For him, the classical rhetorical canons represent not a
collection of skills or even a process, but an ecology of practice situated in a broader ecology of new media (p. 45). The ecology of practice he proposes is the middle layer of new media ecology, sandwiched between an ecology of code about expression and an ecology of culture about social constructivism (p. 47). To support his argument that the rhetorical canon is an ecology of practice Brooke digs into the new media ecology, citing Marilyn Cooper (1986), Margaret Syverson (1999), and Matthew Fuller (2005), George Bateson (1972) and Johnson-Eiola (2005), among many others.

To better explain how the rhetorical canons can operate as an ecology of practice, he maps his ecology of code, practice, and culture on to theorists from the rhetoric and professional communication tradition. His ecology of *culture* is related to Kenneth Burke’s dialectical terms to account for rhetoric, and ultimate terms (Brooke, p.51; Burke 1955, p. 186-187), to account for ideology. He relates his ecology of *practice* to Kaufer and Butler’s representational composition, which is a middle ground between structural composition and genre theory (Brooke, p. 51, Kaufer & Butler p. 5). Brooke considers Clay Spinuzzi’s activity theory-inspired model to be the most similar to his own (Brooke, p. 51). Activity theory, as I explain in chapter 2, is a social theory that distinguishes three layers of activity—one driven by intrinsic motivation, one driven by intermediate goals, and one driven by local conditions. Drawing on Spinuzzi’s *Tracing Genres through Organizations*, Brooke aligns these layers of activity with his ecology of culture, ecology of practice, and ecology of code.

Spinuzzi’s *Tracing Genres through Organizations* expands the notion of user centered design—in particular information design—from an artifact-centered approach to a sociocultural one. He is particularly interested in the “subversive interactions in which workers routinely engage as they use information systems to accomplish their activities,” (p.
4). Rather than seeking to improve design or usability, he seeks to observe the innovative ways workers use technology, finding ways to get work done. His project does for software tools something like what Dorothy Winsor’s project in “Genre and activity systems,” does for documentation. She writes, “Human interaction is apparently never easy. A tool such as documentation is one of the sociotechnical resources that allows the world of the organization to be mediated and maintained,” (1999, p. 222).

One of the important strategies in Spinuzzi’s approach is to take advantage of a concept of “genre ecologies” that he first developed with Mark Zachry (2000). Examining the documentation of two information systems, they observe that workers did not use any official documentation exclusively. Instead, “the technology-in-use…is documented by a perpetually open-ended, dynamic, shifting, and always unfinished ecology of resources encompassing a variety of media and domains,” (p. 170). Genre ecologies exhibit three characteristics: contingency, decentralization, and relative stability. Contingency means that tools are used innovatively and sometimes in ways contradicting their intended design (p. 172). Decentralization means that usability, design, and intention are distributed through the entire ecology (p. 174). Relatively stable means that genre ecologies are consistent enough to be useful but not unchangeable. (p. 175).

Spinuzzi further developed the concept of genre ecology in Tracing Genres, emphasizing how they play a key role in activity systems. He writes, “Any given genre is used to mediate activities in one or more activity systems. But it does not and cannot do the work of mediation all by itself—genres are oriented to different sorts of problems and have developed relatively stable connections or coordination with other genres,” (Spinuzzi, 2003, p. 48). His use of the term “mediation” is important because it indicates that genres are in
part responsible for distributing ideas through a network. More importantly he emphasizes that genres do not work in a vacuum, but are connected and coordinated with each other and, presumably, with the activity of the organization.

Spinuzzi, Hart-Davidson, and Zachry further explore genre ecologies by shifting perspective to “chains” of communication. They write “the chain of communication is a chain of custody of a particular piece of information,” (2006, p. 43). Their insight is that it is not only the genre, or form, of the communication that is relatively stable, but entire sequences of using information are stable (p. 44). Workers facing activities move smoothly from one task to the next, moving information from one genre to the next, in relatively predictable ways.

**Implications of Ecology Metaphors**

Several disciplines have used ecology metaphors to describe and explain tools, activities, and ideas. Ecology metaphors are advantageous because they help appreciate that all three things are interrelated. Ecology metaphors paint a picture of a network of artifacts and actors that has to be understood as a collective whole. It implies a sociocultural approach to usability that is critical for understanding how teams transform loose collections of social and collaborative tools collaborative ecologies. In chapter 2, I will revisit the history of the family of ecology metaphors I have described, tracing them back to their origins in different disciplines. Since that history is intertwined with traditions of usability that part of chapter also provides background on the activity and ideology that shapes the use experience team I studied for this dissertation.
Assembling and Shaping Collaborative Ecologies

Collaborative ecologies do not spring forth like magic, they are constructed. In “Non-academic writing; the social perspective” (1985) Lester Faigley described the emergence of constructivism in professional communication. He writes,

Researchers taking a social perspective study how individual acts of communication define, organize, and maintain social groups. They view written texts not as detached objects possessing meaning on their own, but as links in communicative chains, with their meaning emerging from their relationships to previous texts and the present context. (p. 50)

Social and collaborative software tools can be understood in a similar way. Like texts, software applications are not just functional instruments. When groups of users adopt a software application the software and users become co-constructive. By selecting tools, developing practices, and mediating ideas about those tools and practices through their organization, groups of users transform a collection of tools into a usable collaborative ecology. The selection, practice, and mediation concepts I propose help explain how. This section explains the rhetorical roots of these concepts, and then begins to define them.

Selection, practice, and mediation are fundamentally rhetorical behaviors. In this dissertation, the term “rhetoric” has two senses. One is a very concrete sense that has been understood since classical times, when Plato used the term rhetoric to refer to a kind of speech, related to politics, which produces conviction without producing knowledge(2010, 449a, 463a-e). But Aristotle’s more clinical definition, as “the faculty of observing in any given case the available means of persuasion,” (2007, 1354a) is more useful here. For Aristotle it is an art that can be learned and taught, and it comes in three forms related to civics—forensic, for trials, deliberative, for legislation, and epideictic for praise and blame.
Aristotle’s rhetoric was a set of skills and tactics for gaining influence through speech in the democratic environment of classical Athens.

The second sense of rhetoric arose during the postmodern turn in the middle of the twentieth-century. As the influence of linguistic semiotics (Saussure, 1983) grew, perspectives on language began to change. In rhetoric, language became a vehicle of symbolic power. As Burke explains, rhetoric “is rooted in an essential function of language itself, a function that is wholly realistic, and is continually born anew; the use of language as a symbolic means of inducing cooperation in beings that by nature respond to symbols,” (1955, p. 43). Following this understanding of language and symbols, rhetoric is about articulating power through a semiotic domain. It is an essential force of language that constitutes society. Burke developed a grammar and a rhetoric based on this understanding of language, apart from but continuous with the classical tradition. The foundation of his rhetoric is a grammar of scene, agent, agency, act, and purpose (1945, p. xv) that served as his most famous analytical framework.

The three mechanisms for assembling and shaping a collaborative ecology that I propose are in indebted to both senses of rhetoric. Selection, practice, and mediation are specific activities that play a concrete role in assembling and shaping a collaborative ecology. They are constructive behaviors that help to form connections and articulate power through a collaborative ecology. My concepts of selection, practice, and mediation are indebted to a broad reading of interdisciplinary theory. Selection, for example, is informed by software requirements engineering and traditional usability. Practice is informed by genre theory’s exigency and activity theory’s action, activity, and operation. Mediation is informed by actor-network theory’s translation/mediation. In the next few pages, I define and explain each
of these three rhetorical mechanisms, and explain their theoretical roots more fully in the next chapter.

**Selection**

A selection is a choice that indicates a tool preference. We live in a technological environment saturated with tools performing extensive and overlapping functions. There seem to be dozens of apps for everything. Collaborators make countless decisions regarding tools to purchase, adopt, and employ in countless situations. Since making explicit or implicit decisions about technology is a logical requirement of using a tool, selections deserve to be recognized in particular.

I distinguish two forms of selection. A categorical selection involves making tools available for future use. It is the result of an explicit decision. You might imagine a manager or an engineer identifying functional needs, comparing costs, installing and deploying infrastructure, and consequently making a tool available for future use. You also might imagine a customer walking out of the local computer store with a new antivirus application, or even just downloading a free copy; a categorical selection is made without actually using the tool first.

An instance selection is when a user actually uses the tool. Instance selections show preference for one tool over another, though the user herself may not consciously consider the reasons for her preference. We might imagine a case where only one tool offers the instrumental facility to perform a task, but even then the user could choose to avoid the task completely rather than select a tool. We might also imagine a case where a tool is ostensibly
designed for one purpose, but used for another purpose. Indeed some tools are designed specifically to support improvised solutions to unknown problems.

**Practice**

A practice is a reoccurring use of a tool. Practices include behaviors that follow the tool’s intended design but also unintended behaviors and even avoidance. In that regard the concept overlaps slightly with selection. Practice is distinct from selection in that a practice requires repetition, is concerned with the manner of use, and is interested in the conventionality of a behavior.

Like *selection*, practice can be divided into two categories. An emergent practice is a new pattern of behavior. It may involve only one person, repeating a tool selection over and over. You might imagine a designer who uses an open source image editor rather than the industry standard. Or consider an IT support professional who keeps a wiki page for common questions, and answer emails with a simple hyperlink. For each, the tool satisfies a need well enough to be used repeatedly and to become routine.

A conventional practice is distinct from an emergent practice in that it bears the weight of expectation. The practice becomes not only stable but normative. Co-workers will operate under the assumption that others will follow the convention. A conventional practice can be normative even when the practice applies to just one person. As an example, if we consider the case of the hypothetical IT support person, when coworkers start writing things like “Can you send me that link to your wiki page” instead of “Can you help me reset my password,” it suggests that the support person has a stable and individual conventional practice that is normalizing the behavior of others who do not follow the practice themselves.
Mediation

Mediation is an activity that spreads a conventional practice through a network. Ideas about what tools to use for a task, and how to use them are intrinsically local phenomenon that is particular to individuals and their perspectives. Yet, whole organizations sometimes seem to have uniform patterns of activity. Mediation accounts for the balance between organizational uniformity and local contingencies.

As I’m using it here, the notion of mediation is indebted to Latour’s *We Have Never Been Modern*. Mediation is about extending networks (1993/1991, pg 11). Latour employs the term “translation” as a synonym of mediation, calling to mind the notion of moving ideas from one area of a network to another area. The term “mediation” also calls to mind activity theory’s seminal understanding of tools as a mediating artifact between subject and object. I will endeavor to keep the two senses distinct. As I discuss informal and formal mediation here I mean mediation in the sense of translation from Actor-network theory.

Like selection and practice, mediation can be divided into two categories. Informal mediation spreads knowledge through interpersonal associations and unstructured interactions. Recalling our hypothetical IT support person again, if she casually mentions her improvised F.A.Q. wiki to a colleague, she is mediating informally. But mediation need not be a conscious act. Perhaps the IT support professional picked up her practice when she herself had a question and was referred to a wiki. The tool can evangelize for itself.

A formal mediation is distinct from an informal one because it recruits a wider and more powerful network. Formal mediation typically is intentional. If the IT support person gives a demonstration of her improvised wiki page, or writes instructions, she is engaged in formal mediation. If by chance her manager sees her using it and makes her practice a team
policy, the manager too is formally mediating the technology’s use. Even if the manager forbids it, and threatens disciplinary action to squash the improvised online help, that manager’s act would still fall under mediation since it influences the spread of knowledge. Mediation does not necessarily mean expansion.

**Theoretical Alignments of Selection, Practice, and Mediation**

To conclude my introduction of selection, practice, and mediation as mechanisms for assembling and shaping a collaborative ecology, I want to recognize some underlying theoretical and disciplinary alignments. Selection effects which tools—especially software tools—appear in the collaborative ecology. Selection operates in the material domain and HCI is the discipline with the longest history addressing it. Practice affects the activities people pursue within the collaborative ecology. Practice operates in the behavioral domain and the discipline with the longest history of address it is CSCW or, more generally, social psychology. Mediation affects how ideas move through the collaborative ecology. Mediation operates in the semiotic domain and RPC has the longest history of addressing it (Table 2).

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Artifact</th>
<th>Domain</th>
<th>Discipline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection</td>
<td>Tools</td>
<td>Material</td>
<td>HCI</td>
</tr>
<tr>
<td>Practice</td>
<td>Activities</td>
<td>Behavioral</td>
<td>CSCW</td>
</tr>
<tr>
<td>Mediation</td>
<td>Ideas</td>
<td>Semiotic</td>
<td>RPC</td>
</tr>
</tbody>
</table>

*Table 2: Selection, practice, and mediation alignments.* Selection, practice, and mediation align with particular types of artifacts, domains, and disciplines.
Conclusion

The goal of this dissertation is to examine a user experience team’s activities, collaborative ecology, and the productive rhetorical mechanisms that assemble and shape it. This first chapter outlined a study based on that goal, explored the ecology metaphor, and proposed selection, practice, and mediation as rhetorical mechanisms for shaping and assembling a collaborative ecology. The remaining chapters expand on the project laid out here. Chapter two traces the emergence of ecology to a sociocultural turn in three distinct traditions of usability, building a theoretical foundation for selection, practice, and mediation while providing some background for understanding the activities and ideology of a user experience team. Chapter three explains and justifies the data collection and analysis methods used in my empirical examination of a user experience collaborative ecology. Chapter four tells the story of five threads of activity in a user experience collaborative ecology. The case narrative focuses especially on the tools the team uses to support their activity, and examines it through the selection-practice-mediation framework. Finally, Chapter five identifies some of the key conclusions that can be drawn from the case.
Chapter 2: Theorizing a Rhetoric for Collaborative Ecologies

The previous chapter introduced a perspective on the shared use of social and collaborative tools. It outlined a study of a user experience team’s activity and supporting technology. It explored the ecology metaphors that are now common in sociocultural analyses of technology. Finally, it proposed selection, practice, and mediation as mechanisms that assemble and shape a collaborative ecology. In this chapter, I connect some of these ideas to their intellectual roots in HCI, CSCW, and RPC. First, I explain the artifact-centered beginnings of usability studies. Second, I describe how research has turned toward sociocultural usability. Reviewing this evolution from traditional to sociocultural usability also serves as background for my ethnographic case on the collaborative ecology of a user experience team that shares an interest in usability. Finally, this chapter extends the discussion of selection, practice, and mediation by anticipating some properties of the concepts that can be drawn from existing theory.

Artifact Usability

Over the last few decades HCI, CSCW, and RPC each migrated from an artifact-centric notion of usability to a sociocultural notion. Understanding this evolution requires some background on traditional usability as it has been developed in each tradition for computer tools, separately, for artifacts of communication.

Usability of Computers

The concept of usability is most closely tied to the field of HCI, where it has served as one of the most common ways to measure the quality of design. The strict definition of usability is to improve a product’s “effectiveness, efficiency, and satisfaction,” (Zemel,
Koschmann, LeBaron & Feltovich, 2008, p. 63; Shneiderman & Plaisant, 2010, p. 14; ISO 9241). The concept typically measures ease of use and ease of learning, both of which can be quantified in terms of time (Badre, 2002, p. 5). Shneiderman & Plaisant expand these basic measurements, arguing that the measurements *time to learn, speed of performance, rate of errors by users, retention over time, and subjective satisfaction*, are a desirable framework for design evaluation. Even while traditional perspectives promote advocating users, the important questions are preoccupied with the artifact itself. Does it satisfy users? Is it efficient? Is it effective? How can we improve it?

An early shift in artifact-centered usability is the tradition of affordance beginning with Gibson (1977). While the term “affordance” often employed as a catch-all for functional design features, it actually refers to an external representation of the internal function of an artifact (Norman, 1988, p. 9). A string affords pulling. Strings cannot be pushed. The fact is plain from a string’s outward appearance. Affordances, along with constraints and conventions, are key elements of human-artifact interactions that help a user create a conceptual model of how an artifact can and should be used (Norman, 1999, p. 41). The reason why affordance is important here is because the artifact itself becomes a participant in an interaction with an actual person. The affordances act as the artifact’s language, allowing it to communicate its nature.

Seriously considering the implications of concepts like affordance produces a richer dialogue about the relationship between individual users and their tools. After all, not all users are the same. If different users can have different interpretations of affordances, then design becomes really difficult. This is especially true for systems that are meant to be universally accessible, like emergency resources, which must meet a high rate of usability
across all demographics regardless of “technology variety,” “user diversity,” or “gaps in user knowledge” (Shneiderman, 2000, p. 9). Therefore, the usability of an artifact is particular to the individual user and contingent on potentially large list of factors.

**Usability of Communication**

Conceptually, designing a usable computer system for real users parallels the task of an author addressing an audience. Software addresses users. Writing addresses readers. Both designers and writers peer into the minds of others in order to create successful artifacts. Rosinski and Squire compare the differing approach of each discipline as part of a project to build an HCI-informed pedagogy (2009, p. 150). They point out that while HCI emphasizes participatory design (e.g. Kensing, Simonsen, & Bødker, 1998) and the push for examining real data from real users (Norman, 1999, p. 41), many corners of composition pedagogy consider audiences as imagined or at best minimally involved, following Ong’s “The writer’s audience is always a fiction,” (1975). The imagined audience parallels another usability tradition of developing user “profiles,” that can guide design (Rosson, 2002, p. 2).

Turning from composition pedagogy to technical and professional communication reveals a tradition that more closely mirrors empirical usability and explicitly borrows usability concepts and terminology. Karen Schriver provides a more complete gloss of usability design techniques, pointing out that some are based on intuition and some on research (Schriver, 1997, p. 156-162) and preferring research when feasible. Since her analysis is grounded in the practical need to understand users, it better reflects usability techniques in technical communication as well as HCI.
Because of practical considerations like time and money, HCI designers and technical communication professionals incorporate both hypothetical and user-involved strategies despite the strenuous preference for the later. Over the last few decades, scholars in technical communication have become increasingly committed to examining documents as they are used in real situations. For example, in *User-centered technology: A rhetorical theory for computers and other mundane artifacts*, Johnson theorizes the relationship between humans and systems from a rhetorical perspective (1998). He identifies three models for this relationship—system-centered, user-friendly, and user-centered—that he applies to technology in general and to documentation in particular (p. 122-136). As another example, Grabill and Simmons (1998) see a profound role for usability in risk communication, arguing that technical communicators can "insert the audience/public/citizen directly into the risk assessment process by usability testing" including research practices like "interviewing and observation" (1998, p. 432). In other words, when technical communicators *practice* usability research they also *participate* in the dialog of risk (p. 434), involving the end user more directly.

The concept of usability has been influential in both HCI and RPC for a long time. Understanding how that concept originated and then influenced different disciplines is vital to understanding the later turn towards sociocultural usability and the emergence of ecology metaphors. A secondary benefit is that it helps to inform my study of a user experience team’s collaborative ecology, since usability concepts are influential in their field as well.
Sociocultural Usability

While RPC scholars have continued to develop their own style of human-centered design, a growing camp within the technical communication field increasingly explores technology as parts of sociotechnical systems. Their work parallels and intertwines with the turn towards sociocultural usability in HCI and CSCW. Both RPC and HCI have sought to involve users more directly in design to improve their respective artifacts. But the act of involving users inevitably complicates the equation, leading to increasingly sociocultural approaches. Prominent scholars in this vein include Hart-Davidson et al. (2008), Heaton and Taylor (2002) and Spinuzzi (2008). Others, including Stolley (2009), Ward (2006), Panke and Gaiser (2009), Rice (2009), Sherlock (2009), and Kahn (2000), focus more narrowly on how particular technologies shape the practice of organizations. Applying traditional usability measures is still useful and still helps improve artifacts, but it less useful for understanding the fundamental relationship between users and artifacts. This section first identifies the emergence of sociotechnical understandings of usability. Then, it examines the theoretical systems that have been the most influential for understanding sociotechnical usability, including activity theory, genre theory, and actor-network theory. This discussion provides a foundation to show how each theoretical framework has a particular approach to sociotechnical systems and, consequently, a particular set of analytical advantages.

This recognition comes from Schneider, who points out a divergence between artifact-centric usability where “designers evaluate the artifacts in question by engaging in the study of user interactions,” (2005, p. 449) and distributed usability that takes a “sociocultural turn in our consideration of usability and user-centered design,” (p. 448). Distributed usability, which Schneider traces through Robert Johnson, Clay Spinuzzi, and
others, sees usability not as a property of an artifact but as the product of a sociocultural network. A distributed approach “envisions usability as distributed across the genres, practices, uses, and goals of a given activity,” and can “furnish new insights into the design and evaluation of information systems,” (Spinuzzi, 1999, p.2).

Spinuzzi’s words seem to anticipate Zemel et al.’s question in the 2008 Journal of Computer Supported Cooperative Work:

But where is usability to be located? Is it something built into a product or is it something that can only be found in the emergent practices of the user in interaction with a designed artifact? The definition could be read in either way. If one takes seriously, however, the claim that “technology does not exist independent of its use” (Koschman and LeBaron 2002), then it becomes clear that usability cannot be an attribute of a thing, but rather must be a relation between a user and an artifact as embodied within a set of practices. (p. 63).

The notion that users are involved in producing usability is further echoed by Huatong Sun (2006) in his analysis of cultural usability. Recognizing both Donald Norman’s and Clay Spinuzzi’s observations that end users are ultimately responsible for making a technology work, he writes that users are always actively redesigning, or—more accurately—localizing, an available technology to fit their local contexts. In some sense, who knows users' local culture and contexts better than the users themselves do? Users might not be able to articulate those cultural and contextual factors well, but they know what works in their own contexts, and they know how to make use of a technology in their life spheres if they are able to find a good fit. (pp. 458-459)

In emphasizing users’ roles in designing systems, whether developers incorporate them in the formal design process or not, Sun’s project parallels my own project.

Despite years of developing parallel practices to advance the usability of textual and computational artifacts in HCI and RPC, the underlying connections between the traditions eventually emerged. It shouldn’t be a surprise. Scholarly traditions inside each discipline
have been steadily incorporating sociocultural ideas since at least the 1970s. Of the many brands of sociocultural approaches, the one of the most consistently productive is activity theory, judging by its frequent use in a broad spectrum of research in HCI, RPC, and CSCW.

**Activity Theory Defined**

Activity theory is a popular framework for examining sociocultural systems that proposes cultural and historical links as an explanation for social behavior that seems to contradict individualistic understandings of behavior. Activity theory arose from social psychology, which Voloshinov (1973) defined as the branch of science concerned with “the transitional link between the sociopolitical order and ideology in the narrow sense,” (p. 19). It was first developed in the early twentieth century as a response to behavioral psychology (Cole, 1996, p. 36), and draws on Marxist social theory. Marx believed human consciousness was based on an individual’s material role in society (Cole in Vygotsky, 1978, p. 7). Voloshinov (1973) combined Marxist social theory with Saussure’s semiotics in “Marxism and the Philosophy of Language” to show that speech is profoundly influenced by ideology.

Social psychology is manifested in verbal interaction (Voloshinov, 1973). In activity theory, social behavior arises through the mediating function of signs. An individual achieves some goal outside of her personal ability by recruiting the aid of another person with a verbal sign. The mediating function of verbal signs leads to the development of the “higher psychological functions,” (Vygotsky, 1978, p. 54) necessary for complex social activities. Leont’ev (1978) later proposed a structure for these social activities. Because of the mediating function of signs an “individual consciousness may exist only in the presence of social consciousness and of language that is its real substrate,” (1978, p. 60). By developing
in a society and through time, an individual becomes part of an activity system that can address goals beyond that of the individual alone, motivating the individual to take on new behaviors (p. 63).

From activity theory’s inception, consciousness was a central concept. “Vygotsky described consciousness as a phenomenon that unifies attention, intention, memory, reasoning, and speech,” (Nardi, 1996, p. 11). An individual consciousness interacted with its environment through five key principles, as explained by Kaptelinin, including the unity of consciousness and activity, object-orientedness, the hierarchical structure of activity, internalization-externalization, mediation, and development (Kaptelinin, 1996, p. 107-109).

Unity of consciousness and activity means the human mind (consciousness) “emerges and exists as a special component of human interaction with the environment (activity),” (p. 107). “Mind” or “consciousness” then does not exist independent of environmental features, but develops in response to them.

Object-orientedness means that activity theory considers all properties of an environment, both material and social, to be objective (p. 107). In other words, activity theory doesn’t limit a mind’s experience of environment to mere perception. If a thing looks, acts, sounds, and smells like a duck, it is not only perceived to be duck-like, but functions objectively like duck, and for all intents and purposes is a duck in fact.

The hierarchical structure of activity distinguishes different levels of activity driven by different levels needs (p. 108). At the highest level, activity is self-motivated and when that activity is “frustrated” or interrupted by a change in the environment humans may behave chaotically. Imagine winning the lottery—the self-impelling object for monetary resources disappears, and you go on an irrational shopping spree for boats. At the lowest
level, when the activity is frustrated by a change in conditions, humans automatically change behavior without really noticing. Think of working past sundown, and flipping on a light.

*Internalization-externalization* derives from Vygotsky’s (1978, p. 86) notion of the zone of proximal development, which refers to the range of activities normally outside a child’s capability (external) that can be learned (internalized) through social imitation. As Kaptelinin writes, “mental processes are derived from external actions through the course of internalization,” and “mental processes manifest themselves in external actions performed by a person so they can be verified and corrected,” (Kaptelinin, 1996, p. 109).

*Mediation* refers to human use of tools. “Human activity is mediated by a number of tools, both external (like a hammer or scissors) and internal (like concepts or heuristics),” (Kaptelinin, 1996, p. 109). As explained by Kaptelinin, tool use is culturally situated and develops over time. Mediation is a pragmatic application of cyborgian (Haraway, 1990; Nardi, 1996, p. 7) concepts and is sometimes described by activity theorists as forming a “functional organ.” Kaptelinin writes that functional organs are “the combination of natural human abilities with the capacities of external components—tools—to perform a new function or to perform an existing one more efficiently,” (p. 109).

Finally, activity theory is linked to a notion of *development*. Kaptelinin writes “to understand a phenomenon means to know how it developed into its existing form,” (1996, 109). In other words, the function of a mediating tool/sign is not static but the result of cultural and historical development. That is vital. Or even more succinctly, activity assumes a temporal variable.

Of these principles, the mediating role of tools and signs has been one of the most important, especially in relation to cultural and historical factors. Building on Vygotsky’s
original work, Leont’ev (1981) later added a structure (p. 210), and modern activity theorists often follow Engeström’s (1987) modified version that analyzes subjects who act, the tools they use, the objects they try to achieve, the rules they follow, the community that enforces those rules, and the division of labor within that community (Kuutti, 1996, p. 28). The frequently diagrammed structure (Figure 1) recognizes many the dynamics of social activity.

![Diagram of activity theory](image)

**Figure 1: Formulation of activity theory (From Uden, Valderas, and Pastor 2002)**

In the diagrammed structure, Leont’ev’s recognition that a tool or sign is a mediating artifact that relates a subject to an object is placed in relation to social and organizational dimensions like rules, community, and division of labor. Thus, as artifacts mediate between subjects and objects, so do rules mediate between subjects and community, and divisions of labor mediate between community and object, all in the service of transforming an object to an outcome (Kuutti, 1996, p. 27-28).

**Activity Theory in Workplace Studies**

Activity theory has been consciously employed in sociocultural workplace studies of technology. In the leading chapter of *Context and Consciousness: Activity Theory and Human-Computer Interaction*, Bonnie Nardi articulates the motivation for her now classic
collection of activity theory research. Even though activity theory produced copious and credible theory, a “major American Journal of HCI rejected a set of papers,” for an issue focused on activity theory because of a dearth of empirical studies based on activity theory (Nardi, 1996, p. 5). Her edited collection, *Context and consciousness: Activity theory and human-computer interaction*, and the subsequent two decades of research in the area have more than compensated. Activity theory—understood as the six underlying principles above and as the structured understanding of mediation—has been one of the leading frameworks for understanding social behavior, and as such it has a prominent role in HCI, CSCW, and RPC, forming the theoretical basis for many studies. Examples include Peterson, Madsen, and Kjaer (2002), Hemetsberger and Reinhard’s (2009), Stolley (2009), Nardi, Whittaker and Schwarz (2002), Neale, Carroll, & Rosson (2004), Halverson (2002), and Dayton (2006). This section reviews each one in order to build a further foundation for activity theory and to illustrate its use.

Peterson, Madsen, and Kjaer use activity theory to examine the efforts of two different families learning to use an integrated television and video system, revealing the role of a “process of evolution” and of secondary “learning artifacts,” (2002, p. 102). The role of their secondary learning artifacts seems to parallel the concept of ecologies, since it recognizes that no single technology exists on its own, but is instead intimately connected with and contingent on other artifacts.

In their case study of the open source community responsible for the K Desktop Environment (KDE), Hemetsberger and Reinhardt (2009) employ activity theory to explain the participatory model in a free and open source development environment. They explain that distributed action can be successful through “coat tailing,” which is to “inextricably bind
together individual action and collective activity through careful design of complexes of technological, mental, and cultural artifacts,” (p. 987).

In another study, Karl Stolley uses activity theory in his analysis of the social bookmarking tool Del.ic.ious. In developing Del.ic.ious into a tool to support collaboration among students in a graduate seminar, he identified Del.ic.ious and social media applications in general as open tools supporting action, the middle level in activity theory’s hierarchy. He writes, “SMAs are more constructively viewed when positioned as supporting tool-mediated action in service to a larger activity, such as the team’s documentation revisions,” (Stolley, 2009, p. 356). Following Hart-Davidson (2002), he considers it an “‘upstream’ literate activity…that precedes and supports the activity of revising documents,” (p. 357).

Nardi, Whittaker, and Schwarz (2002) use activity theory to develop their notion of an “intensional network.” They argue that increasingly, collaborative work is defined by networks of personal associations rather than by well-defined teams with stable roles and boundaries. “The bulk of the CSCW literature reflects this view, generally assuming that workers are organized into teams with clearly defined stable roles.” (p. 205). Intensional networks “exhibit aspects of both emergence, being called into existence to accomplish some particular work, and history, drawing on known relationships and shared experience.” (p. 207). Nardi et al. observed the formation of intensional networks, which take on the role of a “collective subject” in a large ethnographic study spanning multiple sites and organization. (238). Nardi et al. name the process of formation “netWORKING.”

Neale, Carroll, & Rosson use activity theory as the basis for their evaluation method for CSCW systems, arguing that “activity awareness” is a vital requirement for working in distributed groups. (p. 115). For them, activities are “substantial and coherent endeavors
directed at meaningful objectives” and that with increased scale, they require a staggering amount of management and overhead tasks (like planning, decision making, and coordination) that “must be understood and pursued in the context of the overall purpose of a shared activity, the goals and requirements for completing it, and how individual tasks fit into the group’s overall plan.” Their “activity awareness” measures how well the system supports individuals’ generalized understanding of all of these elements of the shared activity (p. 115).

Halverson evaluates distributed cognition and activity theory as two frameworks for studying CSCW and for informing design (2002, p. 246). She argues that a theory needs to have descriptive, rhetorical, inferential, and application power (p. 245). Observing that CSCW, and HCI in general, have a tendency to assimilate theory broadly, she argues that the four expectations listed above are a good way to evaluate theories. From these expectations, she argues that “Because of how constructs are named, [activity theory] is perhaps better at supporting discourse within a community that understands the theory, but both [activity theory] and [distributed cognition], like ethnomethodology, have to fall back on the “thick” descriptions of their ethnography to explain their findings to others ‘not in the know’.” (p. 263).

Halverson’s argument echoes Nardi’s own analysis of activity theory, distributed cognition, and a third theoretical perspective called situated action models. She recognizes deep similarities between activity theory and distributed cognition, with the exception of activity theory’s insistence that human agent and artifacts/instruments are fundamentally different (1996, p. 89, 86). She argues that distributed cognition and activity theory are likely to converge while situated action models may not, because “(1) they do not account very well
for observed regularities and durable, stable phenomena that span individual situations and
(2) they ignore the subjective,” (1996, p. 93-94).

Nardi and Halverson compare activity theory to other theories. Dayton, on the other
hand, seeks to hybridize activity theory in his analysis “of work groups as they collectively
learn, analyze, adopt, and redefine a new information technology (IT) tool or system,” (p.
355) at Automated Logic Corporation (p. 368). His complete framework assembled most of
the heuristics of activity theory with the expansive learning cycle and Roger’s organizational
adoption process, with Roger’s adoption and diffusion theory, and with the social
construction of technology.

Dayton’s aim to form a hybrid framework to understand the adoption of innovative
technology parallels my current project, though with two major distinctions. First, my hybrid
framework tends to abstract across the multiple disciplines that inform it, rather than assume
them all wholesale. Second, my case is interested not in the acceptance of a single tool, but in
the dynamic relationships of selection, practice, and mediation that assembles and shapes an
entire collection.

**Activity Theory and Genre Theory**

While research in CSCW continued to develop activity theory as a tool for
sociocultural usability, a parallel effort was occurring in RPC. RPC’s affinity for
sociocultural theory began with the social constructivist turn that Lester Faigley observed in
1985. In the decades since, RPC and rhetoric in general have exploited numerous social
constructivist theories, notably including activity theory, actor-network theory, and genre
theory, its own sociocultural contribution.
By the 1980s, social constructivist thinking had reached a critical mass in the study of non-academic writing. While it existed in parallel with alternative approaches, constructivism was stable enough that Lester Faigley could label it as a dominant paradigm in the discipline that views “written texts…as links in communicative chains, with their meaning emerging from their relationships to previous texts and the present context,” (1985, p. 50).

Social constructivism was a fruitful turn in writing research, and scholars began to search for theoretical frames to support it. The social psychology of Vygotsky (1978), Leont’ev (1978, 1981), and Luria (1976) was just as appealing for the professional communication discipline as it was for HCI and CSCW. For example, Barbara Mirel synthesizes a broad range of social constructivist theories, including the sociology of knowledge, activity theory, the politics of technology, distributed cognition, situation learning, cognitive complexity theory, genre theory, and pragmatic philosophy (Mirel, 1998, p. 18). From these, she articulates themes that apply specifically to writing technical instructions, emphasizing activity in context, experienced problems, and social knowledge.

Among the theoretical frames synthesized by Mirel, the one native to rhetoric and writing is genre theory. While genre has been a concept in writing and literature for ages, it took on new life when scholars began seeing genre as a social phenomenon. Carolyn R. Miller is most responsible for seeding this concept of genre theory in “Genre as Social Action” (1984). As Schryer, Lingard and Spaffod (2007) write, “genres are constellations of regulated and regularized improvisational strategies triggered by the interaction between individual socialization, or habitus, and an organization or field,” (p. 31). In modern professional communication, genres are understood to reorganize speech around
expectations, but they are also “socially enacted” and change over time (Yates, Orlikowski, & Okamura, 1999, p. 70-71).

As inseparable as the concept of genre is from Miller, it is Russian literary theorist and philosopher of language Mikhail Bakhtin who provides the clearest link to activity theory in “Problems of Speech Genres.” Bakhtin defined genres as stable types of thematic content, style, and compositional structure that develop in response to the sphere of communication (Bakhtin, 1986, p. 60). David Russell’s “Rethinking genre in school and society” (1997) recognizes this point of contact between various brands of activity theory and genre theory, explicitly aligning the levels of activity with concepts from Bakhtinian genre theory. He raises genres as tools that mediate stability in individuals’ interactions, (p. 514) and that mediate change in individuals (p. 516), thereby visiting the dichotomy of pattern versus contingency that Winsor (1999) and others continue to point out.

Dorothy Winsor (1999) takes advantage of activity theory in her exploration of the role documentation plays in the dynamic between “regularity and change” in workplace activity systems. She collects theoretical insight from genre theory and she ultimately synthesizes the notion of “activity system” (p. 201) from activity theory with the special treatment of power and agency from actor-network theory (p. 202) to theorize writers’ efforts to “create cooperation in the midst of discontinuity to maintain the coherence of the activity system,” (p. 207).

The combination of genre theory and activity theory continues to be a productive line of research in rhetoric and professional communication in now classic works like Berkencotter (2001), and in more recent discussions of social media like Sherlock (2009). It remains a central theoretical frame even among related and sometimes competing theories
like actor-network theory. For this project, the most important descendent is the “genre
ecology” advanced by Spinuzzi (2003), Spinuzzi and Zachry (2000), and Hart-Davidson et
al. (2008). However, by adding an ecological metaphor, these studies take on new
assumptions about sociocultural activity as discussed in Chapter 1.

To review, the ecology metaphors that are currently productive in analyzing and
explaining the social use of shared tools are a result of a sociocultural turn in the
understanding of usability. As that turn occurred in HCI, CSCW, and RPC, the disciplines
looked to sophisticated social theories to guide their research. Activity theory and genre
theory are two examples that are frequently used together, and both have been productive in
generating insights and scholarly research. Consequently, they offer insights that will
influence and nuance my concepts of selection, practice, and mediation. However, they are
not the only social theories that have been productive. Actor-network theory is a parallel
tradition that developed in the sociology of science, and it too has been productive.

**Actor-Network Theory Defined**

Actor-network theory is a powerful framework for examining sociocultural systems.
Several forms of sociocultural theory appropriate the term network as a metaphor to explain
complex systems of humans, material, and semiotic artifacts, but the one that can most
accurately claim it as a native metaphor is actor-network theory. It was developed by
scholars like Bruno Latour (1993), Michel Callon (1991), and John Law and John Hassard
(1999). In this section, I explain actor-network theory by reviewing several of its most
prominent supporters.
Actor-network theory is characterized as a “ruthless application of semiotics,” that can uncover “relational materiality…and performativity,” (Law, 1999, p. 3-4). As developed by Latour (1993, 2005) and others, actor-network theory sees communities, groups, and institutions as networks or assemblages of human and nonhuman actors. Actor-network theory focuses on how individual actors associate, consolidating and extending power to produce groups, communities, and institutions.

Latour presents this approach in *Reassembling the Social* (2005), where he argues that groups, communities, and institutions should be defined solely by associations among actants. His notion of social is “much wider than what is usually called by that name, yet strictly limited to the tracing of new associations and to the designing of their assemblages,” (p. 7). These assemblages are extremely variable. Actor-network theory resists imposing predetermined frameworks or privileging one actor or network over another actor or network. As explained by Lee and Stenner, “Order is produced over time,” and “Boundaries are not ‘real’ but performed,” (1999, p. 100). No network is a holistic unit. As an analytical tool, researchers merely bound the network wherever it makes sense based on the associations of the participants. Artifacts on the boundaries become particularly important in interpreting the associations between two rival coexisting networks. This is the vision famously explained by Star and Griesemer in “Institutional Ecology, ‘Translations’ and Boundary Objects: Amateurs and Professionals in Berkeley’s Museum of Vertebrate Zoology, 1907-39.” They themselves employ an ecology metaphor to examine how two largely dissociated groups—museum researchers and volunteers—used artifacts of communication as boundary objects. While the object is used in both groups, it is understood differently and in that sense serves a role of mediating between the rival networks.
The actor-network perspective makes social research flexible, but it requires the researcher to accept some controversial consequences. For one, actor-network theory treats human and non-human actors symmetrically (Spinuzzi, 2008, p. 40). Non-human “things” gain a sort of agency in actor-networks (Brown & Capdevila, 1999, p. 39-40). Extending this idea leads to a conclusion along the lines of Herndl and Licona, that agency is distributed through a network rather than resident in a person or thing. In their words, agency “does not reside in a set of objective rhetorical abilities of a rhetor, or even her past accomplishments. Rather, agency exists at the intersection of a network of semiotic, material, and, yes, intentional elements and relational practices,” (p. 137). Documents, events, and other non-humans are treated the same because removing the non-humans from the actor-network makes society “incomprehensible, because its size, its durability and its solidity no longer have a cause,” (Latour, 1993, p 111).

Assigning agency to a network rather than to individuals is a fundamental departure from the more traditional understanding in activity theory, which maintains a strict asymmetry between humans and artifacts (Nardi, 1996, p. 13). The strategy has analytical benefits, but is also famously contentious in that it assigns traditionally human attributes to nonhumans.

**Actor-Network Theory in Workplace Studies**

Just as activity theory has been a productive analytical tool for sociocultural examinations of workplaces, so too has actor-network theory. Notable examples include Latour and Woolgar’s *Laboratory Life* and Winsor’s “Genre and Activity Systems,” as well
as more recent the more recent scholarship of Potts (2005), Rice (2009), and Spinuzzi (2003, 2007, 2008).

Despite its much more recent genesis, actor-network theory has a history of application in professional and workplace environments that is almost as robust as activity theory. A chief example comes from Latour’s own work with Steven Woolgar in Laboratory Life: The Construction of Scientific Facts (1986). Strictly speaking it is not an actor-network theory study. The original printing in 1979 predated the formalization of ANT. Yet it anticipates and informs elements of actor-network theory. For example, the second edition dropped the modifier “social” from the original subtitle, “The Social Construction of Scientific Facts” to indicate the authors’ position that social is a category without meaning (p. 281), which is a conclusion that closely aligns with contemporary actor-network theory.

The book-length ethnography of biologists at the Jonas Salk Institute uses actor-network principles to explain scientific activity—how scientists use instrumentation to generate reality. They write, “Scientific activity is not ‘about nature, it is a fierce fight to construct reality. The laboratory is the workplace and the set of productive forces, which makes construction possible,” (243). Their work is a cornerstone in the social studies of science, a sophisticated analysis of scientific meaning making, and a methodological paragon for a broad collection of sociocultural studies.

Dorothy Winsor was one of the earliest rhetoricians to inject actor-network theory alongside activity theory and genre theory in “Genre and activity systems: The role of documentation in maintaining and changing engineering systems.” Her use of actor-network theory focuses on how actants construct “a network by arranging matters so that other actants have to serve his or her interests to accomplish their own. Actants do so by using
sociotechnical resources, including various forms of written representation,” (1999, p. 202).

This understanding of written texts as part of a network of power was an important tool in her analysis of the constitutional role of documentation in engineering.

Liza Potts used actor-network theory in her analysis of Flickr and the London subway bombings of 2005. In Potts’ words:

Reaching beyond the tracing of a single user experience, actor-network theory (ANT), as discussed below, can encompass an entire ecosystem. Using ANT, researchers are able to look across the mediascape of technologies and people to identify and understand the traces of movements. (2009, p. 285)

She sees Flickr, the photo sharing site, as an assemblage constructed by the combined agency of the camera phone and the human bystander, both being required to transform Flickr from a casual but innovative photo sharing website to a real time clearinghouse of disaster information. (p. 294)

Jeff Rice likewise incorporates actor-network theory in his examination of “networked exchanges” (2009), in particular message boards and blogs. He writes that “networked exchanges distribute writing across authors and places of writing so that the identity of online communicative practice is changed,” (296). In his analysis, a blog page or a message board thread are the artifact produced by these networked exchanges formed by the rhetorical behavior of response.

Of course, Spinuzzi’s “Who killed Rex?: Tracing a message through three kinds of networks,” (2007) is another obvious example. In his analysis, Spinuzzi uses both actor-network theory and activity theory to examine how accounts of an accident that resulted in the death of a pet dog move through a telecommunications company. His goal was partly to examine the case itself, but partly to exercise the distinct advantages and liabilities between
the two theories, to show they are similar in some ways but also distinct in how they understand nonhumans (mediators or actants), how they handle history and development, and their use/disuse of preexisting structure (p.52).

These are just a few examples of scholars using actor-network theory to inform examinations of professional organizations and workplaces. They demonstrate that ANT has particular advantages for addressing certain kinds of questions. For example, questions about power and mediation in large networks are a natural fit for ANT, just as questions that deal with structured activity and history are a natural fit for activity theory.

*Rhetoric for Collaborative Ecologies*

So far, this chapter has reviewed the evolution of usability from an artifact-centered pursuit to a sociocultural one. It was this increasingly sociocultural tradition that produced the variety of ecology metaphors discussed in chapter one. Then, the chapter moved on to review the major social theories like activity theory, genre theory, and actor-network theory that provide the foundation for much of the current sociocultural and ecological examinations of technology. In this section, I again take up the concepts of selection, practice, and mediation that help assemble and shape collaborative ecologies in order to develop them based on the broad reading of interdisciplinary theory and research that I have already reviewed.

While the theories on which I depend are not always compatible, each has a distinct perspective and insight. Consequently, each has particular advantages for addressing certain issues. The selection, practice, and mediation concepts that I propose distill perspectives from usability theory, genre theory, activity theory, and actor-network theory into a tailored
model of how collaborative ecologies are assembled and shaped. The darker dots in Table 3 indicate parts of the model that are of primary concern to the related theoretical perspective.

Table 3: Relationships between selection, practice, mediation, and theory. Darker dots indicate parts of the model that are of primary concern to the theoretical perspective.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Usability</th>
<th>Genre Theory</th>
<th>Activity Theory</th>
<th>Actor-Network Theory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Practice</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Mediation</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>

The theoretical frameworks I have reviewed are useful tools for examining sociocultural phenomenon in general. My project is more narrowly tied to the collections of social and collaborative software employed by groups in the workplace, and I am proposing a model specifically to explain how such collections are assembled and shaped into an effective collaborative ecology. As I wrote in chapter one, the social use of shared tools is a contingent, dynamic, mutually constitutive endeavor. The three constituent parts that I am considering are tools, activities, and ideas. Selection, practice, and mediation assemble them and shape them into a collaborative ecology. Understanding any constituent part requires an understanding of the whole. In the rest of this chapter, I draw on the broad foundation of theory that I’ve collected to further describe and nuance selection, practice, and mediation.

Selection

Selection is the way that software tools are integrated or emerge in the collaborative ecology. Selection is tied to the material constituent of a collaborative ecology. Tools enter
and leave a collaborative ecology as a result of selection efforts. Drawing on previous research about ecologies and usability leads to at least four anticipated properties.

First, individuals select software based on their perception of its functionality, design, and usability. The need for appropriate functionality is obvious, but there is a growing recognition that non-functional attributes like integrity, efficiency, and maintainability also contribute to decision making about software (Chung & Prado Leite’s, 2009, p 368). Usability is the non-functional with most robust tradition, as already described. But most importantly, we know that function of a tool are less important than their affordances, the “perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used,” (Norman, 2002, p. 9).

Second, we can expect software selections to change as the users’ perception of exigencies change. This insight comes from applying ideas from genre theory to software preferences. If we expect “recurrent rhetorical exigence that (brings) together motivations, forms, and audiences to create and sustain” genres (Miller & Shepherd, 2004, ¶2), then we can also expect that the when the exigency ceases to reoccur, the genre will erode or, more likely, evolve. In the same way, as software selections will depend on the exigencies users perceive.

Third, changes in the tools used will be more or less disruptive depending on what level of activity changes with it. This insight comes from activity theory’s three levels of activity—activity, action, and operation—where changes in the activity system at the operation level are addressed unconsciously while changes at the activity level require a fundamental rethinking of the overarching goal. A selection is a change in the activity system that can be related to any level of activity.
Fourth, the selection of tools will have symbolic implications. As with any decision, the tools with which users choose to surround themselves imply a deeper meaning. Kenneth Burke made this observation with respect to terminology, where a simple choice of one terminology over another can radically influence how humans interpret reality (Burke, 1966 p. 45). Similarly, a simple choice of one tool or another has both symbolic and actual implications for the character of a virtual reality.

**Practice**

Practice is the way that behaviors are integrated or emerge in the collaborative ecology. Practice is tied to the behavioral constituent of a collaborative ecology. The tools that users employ—and how they employ them—emerge and are stabilized by developing practices. Drawing on previous research leads to four anticipated properties of practice.

First, the practices that develop in the field may not match the intent of the tools design. There is no evidence that suggests that the actual use of a tool must match what the designer intended, and a host of evidence that suggests it can be different. This is part of the reason for considering Norman and Gibson’s *affordance*, which emphasizes not how a tool is *supposed* to be used but rather how it *could* be used. It is also part of the reason that Spinuzzi points out that users of information systems engage in “subversive interactions…to accomplish their activities,” (2003, p. 4). They redefine the nature of the tools to suit their own needs.

Second, a poor match between available tools and intended activities will result in improvised and innovative practices, or a compromise of activity. A mismatch between activities and tools creates a crisis within a collaborative ecology. That crisis can only be
resolved by changing the tools available in the collaborative ecology, changing how the tools
are used, or compromising the activity. These kinds of mismatches may underlie Spinuzzi’s
“subversive interactions.” We can imagine that the more unusual a tool use appears, the
larger the mismatch between the tool and the activity. This should not imply any sort of
“ideal” tool-activity match can be found, but rather that strange activities may indicate
unstable tools and practices.

Third, practices are performed and not enshrined. As practices emerge and are
repeated that repetition will tend to stabilize them. They can even become expected and
normative. However, the stability, expectation, and normativity must be considered parts of a
performance and not entities of their own. This insight follows a pattern from actor-network
theory, where boundaries between social groups are considered performed (Lee and Stenner,
1999 pg 100), and when they cease to be performed the grouping vanishes (Latour 2005, p.
37). Conventions of technology practice will follow the same pattern.

Fourth, practices emerge and stabilize along a time dimension. Conventions of
practice take time to emerge and stabilize, and can also take time to erode. Humans have
memory, and current character is affected by their past experiences. Arguably, a
collaborative ecology has memory too, or at least it has residual networks of still-stable
conventional practices that linger as the ecology changes around it. This does not mean the
time to stabilize and destabilize conventions is always long. If a barn burns down, the
farmhand won’t be sweeping it in the morning. That practice is done. But he still shows up in
the morning.
Mediation

Mediation is the way that tools and behaviors spread in the collaborative ecology. Mediation is tied to the semiotic constituent of a collaborative ecology. In other words, it is related to people’s ideas and representations of those ideas. Drawing on previous research leads to four anticipated properties of mediation.

First, mediation requires an opportunity. For ideas about a tool or practice to spread, certain conditions must be satisfied. These conditions must arise at the same time and in the same place, or else mediation cannot occur. The particular conditions required for mediation may always be the same, but could be tied to a particular collaborative ecology or a particular region of the ecology. Some conditions might be related to money, time, social awareness, or others. The importance of money is obvious in a corporate context. Time can be involved in developing and mediating knowledge if, for example, special training is required. Social awareness can be involved as well, meaning that residents in one region of a network have to have some symbolic link to another region in order for mediation to occur.

Second, we can expect the psychological principle of satisficing to hinder mediation. Satisficing is a phenomenon recognized in psychology, defined as “choosing decision outcomes that are good enough to suit decision makers’ purposes, but that are not necessarily optimal outcomes,” (Agosto, 2002, p. 17). As I just mentioned, for ideas, knowledge, and beliefs to move through an organization demands that certain conditions be satisfied, but that does not mean it occurs without effort. If a practice serves users reasonably well, satisficing may prevent mediation even if another practice would serve better. Users may not be willing to put effort toward bringing about the conditions for mediation, whatever they happen to be, and may not even be in sufficient crisis to go looking for practices that suit better.
Third, mediation is concerned with expanding and contracting stability. Where practice had to do with stabilizing the use of a tool locally, mediation has to do with how far that tool use spreads. Both are involved in the idea of stability. We can think of the stability that arises from practice as depth, and the stability that arises from mediation as breadth. This follows closely to the original meaning of mediation or translation from actor-network theory (Latour, 1993, p. 11-12), only applied specifically to knowledge about how to use a tool.

Finally, mediation will be tied to implications about agency. Mediating ideas about how to use a tool can involve lots of parties. Certainly, it requires a human with the capacity to create knowledge. Mediation may also arise from the actions of another human, teaching and advocating for a tool use. But, as we’ve seen in the discussion of affordances, tools themselves also communicate ideas about how they can be used (Norman, 2002, Gibson, 1977), and they have a meaningful semiotic contribution to the collaborative ecology’s character.

**Conclusion**

Selection, practice, and mediation are three concepts that are in part responsible for how a group of coworkers transforms a collection of software applications into a successful collaborative ecology. Together they represent a model for how collaborative ecologies are assembled and shaped. Based on reviewing three broad disciplines that are interested in such ecologies I’ve identified some characteristics of selection, practice, and mediation.

The next chapter describes and justifies the methodology I use for a study of a user experience collaborative ecology. Then, Chapter four reports what I find there, with particular attention to the ecology’s activities, the tools that support them, and the work of
selection, practice, and mediation within. Looking at events, quotes, successes, and failures in the ethnographic case demonstrates some of the role of selection, practice, and mediation in constructing the collaborative ecology of a working team of user experience professionals. This process generates a map of the collaborative ecology, and it makes it possible to judge which parts of the collaborative ecology are more or less stable, and to understand changes in tools and practices. It will also help refine the selection, practice, and mediation model I’m proposing.
Chapter 3: Methods

In the last two chapters, I have explained that in the current state of social and collaborative technology research and design in HCI, CSCW, and RPC, researchers need to consider entire collaborative ecologies rather than individual tools and practices. I have also examined the scholarly traditions that lead to this conclusion in each discipline as background for my own ethnographic case and as a source for my own framework for understanding how collaborative ecologies are assembled and shaped. In this chapter, I explain and justify the methods in my ethnographic case study of a user experience team’s collaborative ecology. The chapter is divided into two parts. The first part reviews the theoretical concerns that drive the project’s methodological approach. The second part explains the procedural methods used to conduct the research.

Methodological Choices

The methodology driving this study is founded on two distinct characteristics of the project in general. First, it considers the user to be active participants in shaping the function and design of software rather than simply recipients of it. Users’ rhetorical and social behavior continues shaping software products long after they are released and regardless of any contact with the formal development team. Second, and more importantly, the project is not tied to any particular software but instead examines a complete collaborative ecology, consisting of tools, users, and behaviors. These characteristics lead to four methodological choices that I use to address the dynamic social environments that surround the group use of shared tools. First I take an ethnographic approach. Second, I assume the model of group formation from actor-network theory. Third, I view collaborative ecologies as assemblage of
tools, users, and behaviors. And finally, I use qualitative interviewing as its primary means of data collection.

**Choice 1: Ethnography for Human Centered Design**

This project uses ethnographic methods as an approach to understanding human centered design as *practiced by users in the field*. Ethnography has been the subject of an open conversation, especially in design and human computer interaction, since even today it can be difficult to apply its findings directly to design. Despite its drawbacks, it is a common and effective way to conduct naturalistic inquiry that uncovers concepts meaningful to participants in the field.

The relationship between ethnography and design has been tense at times. On one hand, the participatory design movement has been friendly toward the use of ethnographic methods as an alternative to traditional requirements gathering (Shapiro, 1994) and to gain a deep sense of users’ knowledge and practices (Kujala, 2003, p. 8). On the other hand, pure ethnomethodology resists generalizing and theory building, since anything learned is a social construction that is local to the group studied (Shapiro, 1994), and producing a traditional ethnographic report (Kujala, 2003, p. 5) can be seen as an encumbrance to design.

The methods of ethnographic inquiry themselves, though, are usually regarded favorably. Ethnographic methods have been useful for addressing practical sociocultural problems for at decades. In computer supported cooperative work, they have been seen as an alternative to “traditional requirements capture” since Shapiro’s 1994 analysis of and critique of ethnomethodology (p. 417-418). Despite arguing that purist ethno-methodological research is too focused on description and too antagonistic to building theory that would
benefit software design (p. 418) he still offers it as a useful counterpoint to theory-building traditions. He writes that:

prospects for determinacy and predictability seem to be increasingly thrown into doubt, and with them the prospect for such classically modernist projects as the “rational design” of artefacts, organisations, institutions, political systems, personal relations, or whatever. (422).

One benefit of ethnographic methods is that it can reduce the space between researchers, designers, and users. As user involvement and participatory design have grown in status, efforts to employ ethnographic approaches have increased. These efforts typically follow the sort of hybrid approach introduced by Shapiro. Later research, including Button and Dourish (1998), Kensing (1998), Crabtree (2000), and culminating in Blomberg and Burrel’s 2009 *An Ethnographic Approach to Design*, suggests a rising status for ethnographic inquiry.

Because this study is directed at the question of how coworkers transform a collection of tools into a successful collaborative ecology, it follows in the tradition of ethnographic methods without attempting to create a full ethnographic report. An ethnomethodological approach is warranted because each collaborative group is a distinct, dynamic entity with socially constructed rules and realities. The study is also aimed at the practical goal of informing the design and facilitation of such piecemeal collaborative ecologies, which will benefit from ethnographic insights.

**Choice 2: The Actor-Network Theory Understanding of Group Formation**

One way to address the tension between purely descriptive ethnomethodology and the hope for generalized theory while at the same time respecting the co-constructive nature of technology and social behavior is to assume actor-network theory as an approach to social relationships. Actor-network theory reshapes ethnography because in Latour’s words we
need to think of “sociology as a science of associations” instead of a science of social groups.

He presents this approach in his article, *Mixing Humans and Nonhumans Together: The Sociology of a Door Closer* (1987), and more thoroughly in *Reassembling the Social* (2005), where he argues that groups, communities, and institutions should be defined solely by associations among actors. He devises a notion of social that is “much wider than what is usually called by that name, yet strictly limited to the tracing of new associations and to the designing of their assemblages,” (p. 7). The benefit of this view is that “Actors do the sociology for the sociologists, and sociologists learn from the actors what makes up their set of associations,” (p. 32).

The goal of ethnography in a sociology of association changes. Instead of an attempt to learn the structure of a community, it becomes an attempt to map the network of associations that continually produces the culture. A “community” becomes an assemblage that “…relies on an open flow of communication…between its elements,” (Lee and Stenner, 1999, p. 100). The open flow of communication is observable, and tracing it can reveal the productive efforts of association that are hidden by notions of community, or in this case, an entire collaborative ecology made up of group members, social and rhetorical practices, and a variety of tools.

**Choice 3: Viewing Collaborative Ecologies as an Assemblage**

Under actor-network theory, a collaborative ecology can be understood as an assemblage made of human actors, social and rhetorical practices, many software tools, and other artifacts. Focusing on assemblages to define study boundaries departs from more traditional approaches to ethnography. Traditionally, community is the “focus” of the study,
around which the researcher establishes a boundary between what will and will not be studied, (Miles & Huberman, 1994, p. 25). Communities are defined spatially and temporally (p. 26) because individuals who share a space and time have a common interest in local resources (Shumar & Renninger, 2002, p. 6). Traditional approaches are not the best match for this study for two reasons—distributed working practices and the integral role played by technology.

Many associations in this study are enacted entirely online. Online interaction causes a crisis for definitions of community based on shared material dependency because individuals are involved in meaningful experiences of community that in part a-material (Feenberg & Bakardjieva, 2004). Researchers respond by tying online studies to locational communities (Miller & Slater, 2000, p. 2), by conceptualizing online communities as “virtual” (Feenberg & Bakardjieva, 2004, p. 6), and everything in between. But actor-network theory avoids the crisis entirely because it never relied on spatial and temporal barriers. As Lee and Stenner write, “Order is produced over time,” and “Boundaries are not ‘real’ but performed,” (1999, p. 100). The challenge of bounding a community changes to a challenge of identifying the associations that actors consider meaningful. Distributed working environments enable almost limitless kinds of associations. For example, communities might be bound by shared personal interest (Baym, 1998, p. 38), common practices, (Chandler, Burnet & Lopez, 2000, p. 348), shared virtual space (Meyrowitz, 1985 p.115; McLuhan, 1964; Goffman; 1959), or commercial material exchange (Barney, in Feenberg and Barney, 2004, p. 47) to name just a few kinds.

The second reason to consider assemblages rather than communities is to account for the role that technology plays in shaping the collaborative ecology. Understanding
technology’s role in a collaborative system requires that we avoid viewing it simplistically. Technology is neither a “neutral means” of extending human agency nor a “controlling and determining force” (Verbeek, 2005, p. 11). Instead, we can think of technology as a mediator that actively shapes the relationship between humans and their environment in both directions. Individual technologies become opportunities for interactions between humans and nature. This strategy is indebted to Heidegger’s *Question Concerning Technology*, among other works, that argued against instrumentalist accounts of technology (1977). Taken to its conclusion the strategy leads to a sort of posthumanism where, “our brains and bodies couple to new tools, yielding new extended thinking systems,” (Clark, 2003, p. 197). Actor-network theory pushes this strategy even further by viewing technologies not as “neutral ‘intermediaries’ between humans and the world, but mediators: they actively mediate this relation,” (Verbeek, 2008, p. 114), and is therefore a useful methodological perspective for dealing with sociotechnical systems.

**Choice 4: Qualitative Interviewing as the Primary Means of Data Collection**

The key data collection tool of ethnographic research employed in this study is qualitative interviewing. This project follows the participatory design tradition in human centered design by relying on some ethnographic methods, like qualitative interviewing and participant observation, as a means to understand the social and rhetorical behaviors of a group of collaborative software users.

Transforming a collection of software tools and social conventions into a successful collaborative ecology is an activity tied to a particular group. Every group is a distinct entity with its own needs, concerns, influences, habits, practices, membership, and dozens of other
features, all of which dynamically change over time and space. While analytical tools like the theory of reasoned action and the Technology Acceptance Model (Venkatesh, Morris, Davis, & Davis 2003, p. 426) are useful at quantifying and generalizing elements of the relationship between users and applications, the distinctiveness and dynamism of social groups add a dimension of complexity that is particularly difficult to address, and benefits from qualitative inquiry and sociocultural examination with tools like activity theory (Kaptelinin, V. 1996 p. 107-108) and actor-network theory (Law, 1999, p. 3-4).

This project uses an approach based on qualitative interviews. Qualitative interviewing offers several advantages for examining problems that are linked to particular groups of people. In particular, it facilitates developing detailed, holistic descriptions, describing processes, and learning how events are interpreted from an insider’s perspective (Weiss, 1994, p. 9-10). Through immersing in a culture and recruiting informants, it becomes possible to understand the local response to the challenge of transforming a collection of software tools and social conventions into a successful collaborative ecology.

Qualitative interviewing ties the results of this project to the particular people, tools, practices, and ideas in the examined assemblage. The participants’ responses to the social construction of collaborative ecology, their interpretation of events and their shared meanings will remain local. Still, patterns of behavior and meaning are likely to emerge. These patterns can be compared against previous social theory and can be used as a basis for future work in this area. Similarly, many elements of the approach to learning about this local problem will be applicable to other local problems.
Procedural Methods

Part one explained the broad methodological decisions that shape my research into the social construction of collaborative software environments. This part explains and justifies the procedures used to conduct the research itself. These include criteria for selecting a research site, descriptions of the research site, guidelines for interacting with study participants, for collecting data, and for analyzing that data.

Research Site Requirements and Description

Like any ethnographic study, the findings of this study are specific to the people and environments examined, and will not universally generalize to other sites. However, the members of that site participate in issues that are not uncommon, and the insight gained from the study can inform our understanding of other contexts. To examine challenges that are germane right now, in the context of distributed workplaces, mobile computing, social software, and all the other innovations pouring fourth from app markets and digital storefronts, it is necessary to impose some criteria on the selection of research sites. Here, I identify five that guided my selection the site, including group size, the presence of virtual or distributed collaboration, the incorporation of modern information technology, residence in a workplace setting, and the possibility of gaining research access.

Some of the most interesting features of modern social software are most evident at an extremely large scale. For example, flash mobs (Shirky, 2005), niche markets (Anderson, 2006), and crowd wisdom (Benkler, 2006, Suroweiki, 2005) are all side effects of huge numbers of people coordinating behavior through technology. On the other hand, interesting things happen in the world of just a few individuals (Walther 1996, Walther et al 2009, Tong et al 2008). Still, much professional work relies on smaller teams of roughly between 5 and
25 individuals. This size corresponds to the size of an individual's “primary network” of close companions in an anthropological sense (Dunbar 1983 p 681-682). The haphazard interconnections between many such small networks is part of what makes large networks such efficient vehicles for moving information (Shirky, pg. 221). Because of the unique features of small groups, research sites that consist of groups roughly between 5 and 25 members were considered favorably in this project.

In this project, I needed a research site that included at least some virtual collaboration. For successful teams, the social construction of a collaborative ecology must logically occur regardless of the physical collocation of the workers. Still, teams that are wholly or partially virtual face new and unique challenges, (Jarvenpaa & Leidner, 1999 ¶ 8; Paretti, McNair, & Holloway-Attaway, 2007, p. 327-328; Townsend, DeMarie, & Hendrickson, 1998, p. 22). A prominent example is the means individuals use to form and present identities (Vie, 2008, p. 21; Zywica & Danowski, 2008, p. 3) and how they are understood by others. These kinds of challenges and the tools and behaviors employed to address them are part of what makes collaborative and social software an interesting field of study. In order to explore these challenges and behaviors, this study required a research site that was at least partially enacted using innovative communication technologies.

Taking on a distributed research site did cause some difficulty for data collection. Qualitative interviewing and field observation often required virtual presence, since physical collocation was typically impossible. Furthermore, remote collaboration was common among the participants of the study; a few members work from a distance almost exclusively. By conducting some interviews and observations virtually, the research experience reflected the typical working experience of the participants.
Aside from including a distributed workforce, this project needed a team that incorporated some novel information technology like social media applications into their working practices, since this study is interested in how coworkers socially construct a collaborative ecology by incorporating a variety of tools into their working behaviors. While the newest and most innovative social tools are interesting, mundane technologies like email and instant messaging remain important, because they also contribute to the character of the actor-network. Consequently, the research site was not chosen based on the use of a particular technology (email, instant messaging, social networking), and certainly not based on a particular instance of those technologies (Gmail, Google Talk, Google+). In other words, the study is bound to the technologies with which the group of coworkers associates and is not to a particular technology. Instead, technologies are treated as constituents of the collaborative ecology alongside individuals and practices. While the particular technology employed was not a factor in the choice of research site, I did seek a research site that incorporated some forms of collaborative and social software, and I preferred a site with innovative software and innovative uses of software.

This project required a workplace setting. Collaborative software applications, especially high profile social media applications like Wikipedia, Facebook, and YouTube, are popular research topics. Much of this research tradition is grounded in personal, (e.g., Walther, Van Der Heide, Hamel, & Shulman, 2009), entertainment (e.g. Sherlock, 2009), and educational uses of the technology (e.g. Moxley, 2008; Stolley, 2009). The insights drawn from these lines of research are intrinsically interesting and applicable in working contexts. This project joins studies of social media applications that are grounded in professional and business contexts (e.g., Zhang et al. 2009; Roberts & Roach 2009; Genova, 2009), which are
somewhat less common. Consequently, this study is devised with business, corporate, and industrial settings in mind.

Selecting a place of business as a research site has its own challenges, particularly in regards to access. Businesses have legitimately heightened concerns for privacy, confidentiality, and the costs of research partnerships, and therefore require increased effort, assurances, and accommodations on the part of the researcher before granting access (Weiss 1994 p.19).

Gaining access to the workplace environment is a challenging prerequisite for conducting a field research study. To find appropriate research sites that fit the criteria above, the researcher leveraged professional social networks heavily. By making contact with professional groups, speaking at industry meetings, and developing professional contacts through working relationships, the researcher found several organizations that were satisfactory for conducting the project.

Based on the study’s needs regarding group size, distributed working practices, incorporation of innovative collaborative tools, and site access, I chose a mid-sized usability and user experience team of roughly twenty professionals working at a multinational corporation as my research site. The team consisted of writers, visual designers, human factors engineers, user experience and usability researchers, interface and interaction designers, software developers, and project managers. The members of group are distributed across four physical working locations. While some collaboration occurs face to face at each location, most tasks require collaboration from all parts of the team. As such the team relies heavily on a wide range of collaborative technologies to coordinate their behavior. The group is relatively stable, and is fixed by a traditional hierarchical organization. Yet because of the
nature of their work, team members frequently partner with professionals outside of their own team, making it difficult to precisely identify the number of members.

**Relationship to Participants**

Establishing a fruitful relationship with participants is a primary concern of ethnographic studies and a prerequisite to collecting reliable data (Creswell, 2007, p. 132; Weiss, 1994, p. 33-34). Establishing a good working relationship in this study meant negotiating the terms of research access, identifying the group membership, recruiting participants, building rapport, and making arrangements for protecting their privacy and confidentiality. In the next several pages, I describe how I addressed each of these concerns in order to improve the quality of the data I collected.

The first task after identifying a satisfactory research site was to negotiate the terms of access for conducting research at the site. The negotiation required extensive discussions about privacy, human resources, confidential and proprietary rules, and intellectual property. Respecting the needs of the members along these lines required limiting the time, duration, and form of data collection, concealing the exact identity of the corporation, its locations, and its trademarks and other intellectual property, and also concealing the identity of individual participants.

After gaining approval to conduct research, the second task in the project was to identify members. I took some cues from the existing organizational hierarchy of the team, but because the team was geographically distributed and because team members typically work on projects outside the team itself, the official organizational charts were only minimally useful in identifying membership. This comes as no surprise in a study that is
methodologically grounded in actor-network theory, which considers official organizational structures only in so far as they carry meaning for individual actors. What is important is association, not organizational structure. Latour (2005) uses the metaphor of the insectoid ant to explain the approach of the ANT researcher, near-sightedly following the trails of local associations rather than trying to predict the overall structure of a community (p. 5).

Because the project did not use a priori membership criteria, and instead followed the interpersonal associations enacted by the team members through word of mouth sampling and by shared activities like meetings. I began with a core group of participants who frequently interacted in work activities and used “snowball” sampling (Weiss, 1994, p. 25) to recognize the participants’ own assessment of important community members. The strategy created the possibility of enrolling study members outside the research site proper, which did occur. Also, no one referenced or interviewed in the study was a member of a vulnerable population, such as minors.

After identifying study members, the next necessary activity was to establish and maintain rapport with members in order to learn their meanings and their interpretations (Creswell, 2009, p. 175). The heavy use of distributed work complicates access and rapport, (Markham, 2005, p. 801), but they remain a necessary part of “getting it right” or at least “try[ing] to not ‘get it all wrong’,” (Miles & Huberman, 1994, p. 277). In naturalistic studies, not getting it all wrong depends largely on the researcher’s relationship to the members of the culture they seek to understand, (Miles & Huberman, 1994, p. 6).

My relationship to members began as that of an outsider. For most participants, this relationship changed through protracted contact during time spent on-site. For seven months I interacted with the members of the team on a daily basis, eventually taking on more of an
This closer relationship facilitated trust and encouraged members to participate faithfully (Creswell, 2007, p. 138-139). Of course, the distributed nature of the community created a special challenge in this regard, but as Hine argues “Whether physical travel is involved or not, the relationship between ethnographer, reader and research subjects is still inscribed in the ethnographic text,” (2000, p. 46). Consequently, the data reflects differences between participants I met in person and those I met only online, but the same differences arise among the participants themselves, too, in so far as their relationships are virtual and remote.

In order to protect the privacy and identity of members as well as possible, this study complied with the formal structures of informed consent and institutional review. Participant enrollment in the study occurred at the beginning of the first interview with each participant. The researcher verbally explained all participants’ right presented the formal letter of informed consent (Appendix A).

In this case, informed consent included the voluntary nature of the study, the fact that it was for research, that participation was uncompensated, and that the participant’s identity and personal information would remain confidential. All publications of data and analysis use pseudonyms for any potentially identifiable information. All voice recordings, transcripts, notes, and any other collected documents were stored digitally under password protection.

Because this research project involved businesses and corporation, the researcher addressed further concerns with regards to intellectual property, proprietary and confidential information, and human resources. Employees were not interviewed on company property, and working hours were not consumed by any research activity. The researcher took care to
Data Collection

Because it is an ethnographic and qualitative interview study, the project depended on site observations and a series of interviews with team members. Interviews took place in conjunction with a seven-month immersion experience with the primary site. In total, the researcher conducted nine interviews, each spanning roughly 60-90 minutes. Interview participants represented a variety of physical work sites, organizational roles, and educational backgrounds (Table 4 next page). Interviews were conducted according to a loosely structured interview protocol (Appendix B), recorded to digital audio files, and then transcribed to facilitate analysis. The next several sections describe the procedures used for each activity in more detail.

To inform the interview process, the researcher prepared a protocol for conducting interviews. The protocol was intended as a guide to manage a conversation rather than an instrument to elicit narrow responses. The questions were open-ended and designed to generate follow-on discussion (Fontana & Frey, 2000; Weiss, 1994). By giving more freedom to the participant, open-ended interviewing allowed for a richer data set and a more faithful representation of participants’ meanings and interpretations, but it also creates challenge in analyzing and interpreting the data.
Table 4: Summary of study participants

<table>
<thead>
<tr>
<th>Pseudonym</th>
<th>Organizational Role</th>
<th>Background</th>
<th>Site</th>
<th>Interview Date</th>
<th>Duration (hr:min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carol</td>
<td>Visual Designer</td>
<td>Graphic Design</td>
<td>Site 1</td>
<td>12/21/2011</td>
<td>0:54</td>
</tr>
<tr>
<td>Jim</td>
<td>User Centered Design</td>
<td>Human Factors</td>
<td>Site 1</td>
<td>12/22/2011</td>
<td>1:05</td>
</tr>
<tr>
<td>Steve</td>
<td>UI Designer</td>
<td>Psychology</td>
<td>Site 2</td>
<td>1/6/2012</td>
<td>1:23</td>
</tr>
<tr>
<td>Alice</td>
<td>UI Designer</td>
<td>?</td>
<td>Site 1</td>
<td>1/10/2012</td>
<td>1:00</td>
</tr>
<tr>
<td>Ray</td>
<td>User Experience Designer</td>
<td>?</td>
<td>Site 3</td>
<td>1/30/2012</td>
<td>1:11</td>
</tr>
<tr>
<td>Tom</td>
<td>UI Designer &amp; Usability Engineer</td>
<td>?</td>
<td>Site 2</td>
<td>1-31-2012</td>
<td>1:02</td>
</tr>
<tr>
<td>Cathy</td>
<td>Customer Feedback Specialist</td>
<td>?</td>
<td>Site 1</td>
<td>3/2/2012</td>
<td>1:06</td>
</tr>
<tr>
<td>Reggie</td>
<td>User Experience Imperative Lead</td>
<td>Programming</td>
<td>Site 1</td>
<td>3/5/2012</td>
<td>1:14</td>
</tr>
<tr>
<td>Sean</td>
<td>Usability</td>
<td>Experimental Psychology</td>
<td>Site 4</td>
<td>5/1/2012</td>
<td>1:33</td>
</tr>
</tbody>
</table>

Interviews always began with a review of the informed consent document and an overview of the purpose of the study. Then the researcher moved through the topics from the interview protocol starting with the participants’ role in the organization. Each interview lasted roughly 60 minutes. Each interview was recorded, with permission, and then stored for later transcription and analysis.

All interviews were recorded on audio and then transcribed to electronic documents in order to facilitate analysis. The researcher used word-for-word transcription of both the interviewer’s and participants’ utterances. Original phrasing and grammar were maintained.
in transcriptions, which include only minor revisions to accommodate the loss of verbal cues. Quotes used for the research report in chapter four have been lightly edited for clarity.

The immersion experience consisted of three months of daily on-site interaction with the user experience team and four months of daily remote interaction. During this time, the researcher was not able to take detailed field notes on account of agreements with the research site. As a consequence, ethnographic observations are reported as personal reflection.

Since this project is primarily concerned with social and rhetorical behavior rather than linguistic features, the transcriptions do not record features like inflection or pauses in the transcription itself. While these features would be considered in many studies of discourse analysis and pragmatics (Schiffrin, Tannen, & Hamilton, 2001; Edwards & Lampert, 1993), this study is more centered on the ideas and knowledge of its participants. However, linguistic turns were maintained as a way to facilitate coding and organization of the data.

Data Analysis

Transcribing interviews was the first step in a comprehensive analysis of the study data. After transcription and subsequent reviews of the data, the researcher began more formal data analysis procedures. The goal of this analysis was to organize the data, identify patterns of social and rhetorical behavior, especially in regards to the tools, and to identify evidence to support an interpretation of those behaviors. The analysis procedure began with selecting a qualitative data analysis package, developing and applying a coding scheme, and
then using these tools to draw insights from the data. The next few sections will explain each activity in more detail.

The researcher collected interviews and transcribed them in RQDA. RQDA is a qualitative data analysis library developed on the R programming language for statistical analysis. It is a free and open source, platform-independent data analysis solution with standard coding functionality and a streamlined interface, for which reasons it was chosen over more command sophisticated applications like Atlas.ti (Barry, C., 1998; Lewis, R.B., 1998), NUDIST, and NVIVO9.

Once transcriptions were collected in RQDA, the researcher began applying an initial coding scheme based on the prominent activities of the study members. The preliminary scheme included seven codes (Table 5 next page). The seven codes are based the prominent activities that emerged in observations and interview. They include a master narrative around the software development lifecycle, software requirements and user data from usability studies, specifications and high-level designs for planning and prioritizing new product features, prototypes and designs for new versions of software products, tools and activities for coordinating work, tools and activities for carrying on real-time collaboration, and an extra code on reports of interpersonal relationships. The initial codes were the first step towards what Weiss calls “mini theories” or a hypotheses that “make(s) sense of material dealing with specific issues” within the data (1994, p. 158), and were eventually redistributed into the five narrative threads reported in chapter four. The material coded under Requirements and User Data was ultimately split into two, while coordination and real time collaboration were distributed into the other categories. The data coded for interpersonal relationships were not used in the study.
Table 5: Preliminary coding scheme and associated narrative thread

<table>
<thead>
<tr>
<th>Preliminary Code</th>
<th>Definition</th>
<th>Thread</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements And User Data</td>
<td>Tools And Practices For Identifying And Prioritizing New Product Features</td>
<td>Identifying And Prioritizing Requirements, Collecting And Reporting User Data</td>
</tr>
<tr>
<td>Specifications And High-Level Designs</td>
<td>Tools And Practices For Writing And Circulating Specifications For New Features</td>
<td>High-Level Designs And Specifications</td>
</tr>
<tr>
<td>Prototypes And Designs</td>
<td>Tools And Practices For Developing Designs And Prototypes For New Software</td>
<td>Design And Graphic Prototypes</td>
</tr>
<tr>
<td>Coordination</td>
<td>Tools And Practices For Coordinating Work, Generally</td>
<td>Integrated Into Other Threads</td>
</tr>
<tr>
<td>Real Time Collaboration</td>
<td>Tools And Practices For Real Time Collaboration</td>
<td>Integrated Into Other Threads</td>
</tr>
<tr>
<td>Relationships</td>
<td>Comments On Interpersonal Relationships</td>
<td>Not Discussed</td>
</tr>
</tbody>
</table>

Each category was used to mark a set of language features in the transcribed data. Codes were applied at the phrase level rather than by the utterance or linguistic turn in order to accommodate the highly variable length of turns. Overlapping codes were permitted.

**Conclusions**

This chapter outlined the ethnographic methods used in the study. It is a research study using open ended, qualitative interviews and ethnographic observations in order to examine social and rhetorical processes that assemble and shape a user experience team’s collaborative ecology. The researcher began by picking a research site to fit criteria like mid-size teams, use of distributed workplaces or virtual teams and reasonable incorporation of currently innovative technology. The researcher collected data by conducting interviews with members of the site and participating in a seven month immersion experience with the
research site. That data was coded for seven initial categories that eventually led to five narrative threads. The next chapter reports those five threads with particular attention supporting tools and to the role selection, practice, and mediation play in assembling and shaping the collaborative ecology.
Chapter 4: A User Experience Collaborative Ecology

The previous chapter described a field study to examine the social construction of collaborative ecologies. This chapter reports the findings of my seven-month ethnographic inquiry into the construction of one user experience team’s collaborative ecology. The story that emerges has five threads of related activity, each one supported by its own collection of software tools (Table 6). It starts with the software development lifecycle (1), which is the most prominent organizing concept influencing the user experience team’s collaborative ecology. Then, the story moves to the activities that collect and report the user data (2), which is the distinct responsibility of the user experience team. The user experience team then injects these data into activities for identifying and prioritizing requirements (3); designing graphic, interface, and interaction prototypes (4); and authoring high-level design and specification documents (5). By retracing these five threads illustrates the relationship between the shared activity of the organization, the individual members, and the collections of software tools that support them.

Table 6: Emerging threads of activity in the user experience collaborative ecology.

<table>
<thead>
<tr>
<th>Thread</th>
<th>Activity</th>
<th>Supporting Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The Software Development Life Cycle</td>
<td>Formal tools that structure the overall organization’s activity</td>
</tr>
<tr>
<td>2</td>
<td>Collecting and Reporting User Data</td>
<td>Tools for collecting, analyzing, summarizing, and reporting user data</td>
</tr>
<tr>
<td>3</td>
<td>Identifying and Prioritizing Requirements</td>
<td>Tools for determining, publishing, and advocating for user requirements</td>
</tr>
<tr>
<td>4</td>
<td>High-Level Designs and Specifications</td>
<td>“Fleets” of informal and open tools associated with the production and circulation of high-level designs and specifications</td>
</tr>
<tr>
<td>5</td>
<td>Design and Graphic Prototypes</td>
<td>“Fleets” of informal and open tools tied to the production and circulation of prototypes and designs</td>
</tr>
</tbody>
</table>
Thread 1: Supporting the Software Development Life Cycle

The most important collaborative tool in the user experience team I studied is a large-scale web application for coordinating the software development lifecycle. For now I will call this package the Agile Manager, since it was designed specifically to support the Agile methodology of software development. As described by the user experience team, Agile Manager is really a suite of integrated tools for managing software project artifacts and coordinating teams. A partial list of its function includes:

- prioritizing software requirements
- tracking their progress through iterations of development
- tracking software bugs and their resolutions
- version control for computer code
- file management for design documents and prototypes
- notifications for task additions and completions
- forums for discussion
- a tagging system to categorize project resources by team

In other words, Agile Manager is a “one-stop tool for managing an entire development lifecycle in order to support the agile process,” (Steve). The Agile Manager tool is a representation of the Agile methodology that underlies its design and is therefore crucial to regulating the user experience team’s role in the software development lifecycle.

The user experience team’s role is to inject a user’s perspective into the software development lifecycle. Various members of the team describe their work as creating studies and collecting user data, identifying and promoting software requirements from the users’ perspectives, developing high-level designs and specifications to satisfy those requirements, and creating graphic, interface, and interaction prototypes to guide programming and ongoing user studies. These four activities each depend on a different subset of the broader tool ecology, which will be described later in this chapter. In one way or another, Agile
Manager brings all these activities together. Agile Manager represents the formal pipeline for driving changes into the team’s software product.

Agile Manager has a privileged place, but the organization actually supports two major software development lifecycle managers. The other, Version Control Suite, is a much narrower tool that has been a key element of the organization’s collaborative ecology for decades. On paper, Agile Manager does everything Version Control Suite does and more. Ever since a corporate purchase of the vendor that developed Agile Manager, Version Control Suite has been in a “phased out” process (Reggie) because even though “[Version Control Suite] has features and other things, but it’s just old, so Agile Manager really is meant to replace it” (Reggie). In the words of Sean, “we eat our own dog food.” Several user experience team members welcome the move towards Agile Manager (Steve), but not every project manager adopts the Agile-friendly application. Version Control Suite is very much alive. As Steve states, “it’s not a simple flip-a-switch scenario” to move to a new management tool, because existing or legacy products have a “huge investment in their existing tool.” Switching involves set-up efforts by administrators, new learning curves for end users, and extra effort to accommodate legacy software builds that are already set up (Steve). These reasons explain why Sean reported “push back” in response to the transition away from Version Control Suite.

Having two software development lifecycle managers supported indicates a tension that traces back to the ongoing appropriation of Agile methodologies. Agile is founded on the idea that initial requirements are always wrong, according to Jakob Nielson (2008, ¶1). There are many reasons, as explained by Cao & Ramesh, who write “Rapid changes in competitive threats, stakeholder preferences, development technology, and time-to-market pressures
make pre-specified requirements inappropriate,” (2008, p. 60). Therefore Agile and other rapid prototyping models use iterations of small compartmentalized improvements, allowing for frequent adjustments to the direction of the project. The Waterfall model, on the other hand, conceives of software projects as a sequence of activities like setting requirements, design, implementation, verification, and maintenance. As Tom says, in Waterfall “you have a big list of stuff and it’s sometimes hard to tell if you’ve met the requirement or not” (Tom). The choice of software development life cycle manager is just one prominent way that the switch to Agile has influenced the user experience team’s collaborative ecology.

The phase-out of waterfall and Version Control Suite is far from complete, and most user experience team members seem comfortable switching between it and Agile Manager. The formal and informal attempts to spread the use of Agile Manager have not produced organization-wide consistency. In a sense, the organization is reliving a phenomenon that first occurred when it switched from a mainframe-based electronic mail system to a more modern system years ago. Then, as now, worker complaints kept two systems operating simultaneously, up until the organization finally “pulled the plug” on the older tool (Sean). That has not yet happened for the software development life cycle management software. The coexistence of the two tools shows lack of mediation and one consequence is emergent interstitial practice.

Supporting two software development lifecycle management tools has required a patchwork of new tools and practices. Individual project managers have the authority to select one or the other, but sometimes separate projects need to interact. For example, a team working on the base edition of an application might be using one suite, and a team working on special features for an advanced edition might be using another. Jim reported that in order
to keep each team aware of the other’s progress, a special group in the corporation developed “bridges” between the two management tools. For example, with these bridges running a new task in Agile Manager is copied in Version Control Suite, and vice versa. The emergent practice and custom tools illustrate some of the consequences of the incomplete mediation of the newer software’s use.

A more prominent consequence to supporting two management tools is an individual need to be flexible with practices. This is especially true for the user experience professionals in my study, who typically operate on multiple projects at once. One day they may be following the practices of Agile Manager to contribute to one project, and the next day they may be following the practices used in Version Control Suite to contribute to another. This instance selection is largely dependent on which tool the team they happen to be working with has selected as their project management tool of choice, which is contingent on the whim of the management and executive classes. Study participants generally felt they had little recourse other than follow the rest of the team (Ray), and that attitude trickled down to affect their informal and open tool practices as well. These minor adjustments are almost akin to linguistic code switching of bilingual conversation—“the alternating use of two or more ‘codes’ within one conversational episode” (Auer, 1998, p.1) and aligns with activity theory’s change in “Conditions” that automatically and seamlessly results in a change in “Operations.”

Working on multiple teams with different collaborative practices is challenging for user experience team members. Some see it as a major inconvenience, like Jim who writes, “Because I work on so many teams and each team, the team leaders or whoever is working on those teams make different decisions then I’m torn into 15 pieces,” (Jim). Others see it as
minor, like Ray who says “it’s not something that’s insurmountable. I would say yeah of course it takes more time but it’s not something that takes so much more time that it’s a huge issue,” (Ray). Either way, the user experience professionals take it for granted that accommodating either practice falls on them as individuals.

The user experience team has a few formal resources to help them accommodate the discontinuities caused by the dueling management tools. One is a page of instructions on a wiki to help new users gain access to their projects in Agile Manager, which is a particularly complex task (Alice). Another is a document indicating key features and offering basic tutorials of the suite (Alice). But largely, participants relied on trial and error and learn-by-doing (Alice; Ray), and on seeking help from nearby coworkers (Tom; Alice). It seems likely that, because the need to switch frequently from one management tool to another is mostly limited to a small group of user experience designers, informal mediation behaviors are preferable to formal ones.

Having two simultaneous suites relying on informal mediation may also contribute high variability in local collaboration practices. For example, Tom described switching learning to use Agile Manager but resisting some of its file sharing features. Instead, he continued operating the way he learned to under Version Control Suite, by storing files on his local machine and sending them through email, until the development team he was supporting reached a critical level of confusion of which file was most current. At that point, he finally adjusted his file-sharing practice to match “what everyone else was doing” (Tom). His eventual change in practice is consistent with his earlier observation about Agile Manager:
I’m not sure we've established best practices for how to use it, and that's actually something I’ve been meaning to bring up at one of our team meetings, because I’ve just been sort of, I’ve been figuring out what to do with it on my own, um, I guess I don’t really know the answer, I think um, it was probably mainly developed with developers, testers, and project managers in mind.

(Tom)

Other user experience team members, in particular Cathy, seem to avoid Agile Manager completely if they can. As a user feedback specialist, her work does find its way in to Agile Manager, but she tends to rely on her colleague Jim to interface with the Agile Manager. These variations in local practice surrounding Agile Manager seem to arise partly from the overlapping functionality of Version Control Suite, partly from the local and informal mediation strategy, and partly from a generally destabilized convention of practice for user experience professionals.

The user experience team’s participation in the ongoing effort to construct a collaborative ecology around Agile development methodologies illustrates several important findings. So far, we have only seen a tiny slice of the collaborative ecology working in the broader organization, illustrated in Figure 2 on the next page. That collaborative ecology includes Agile Manager, Version Control Suite, the bridges between them, a couple of wikis and other websites with instruction and tutorials, and client-side storage solution for file-sharing, and a glimpse of an email system. This diagram and the others in this chapter arrange the tools in accordance with how the members of the team understood the tools’ function. Lines represent associations revealed by team activities. Boxes represent team member’s understanding of the usefulness of the tools. Since team members can maintain more than one understanding of a tool, the same tool can appear in more than one place on the map. Agile Manager seems most central, and the collection of tools and practices that
surround it are in constant negotiation, due in part to broad changes in the organization like the move away from *Waterfall*, but also due to local variations and personal preferences like having to learn Agile Manager by trial and error.

*Figure 2: Tools supporting SDLC management.* The user experience team used two bridged project management tools to support their software development lifecycle, but also needed basic communication tools and file sharing and storage.

Even in this narrow glance at the user experience collaborative offers sufficient evidence to illustrate selection, practice, and mediation. For example, categorical selection can be seen as the broader institution continues to support two management systems, and yet project managers still select which to use for their individual projects. Tom’s ability to choose between either storing files in Agile Manager or maintaining his older habit of sending locally stored files through email is yet another example of instances of selection. As he repeatedly makes the same selection, storing everything in Agile Manager, his behavior becomes a personal practice. The practice of switching from one software development manager to another, and the understanding that it is the individual’s responsibility to learn
how, is an important conventional practice. Tom’s desire to begin a conversation with his team about “best practices” for Agile Manager is an attempt to stabilize practice to get the most out of the tool, and for formal mediation, to help make the user experience team’s use of the tool more uniform. We also see informal mediation, as Alice walks across the hall to Reggie to ask questions about Agile Manager. Building bridges between Agile Manager and Version Control Suite would seem to be a technical solution to compensate for a problem in mediation, enacted at an institutional level. The bridges reduce some of the pressure to have a uniform practice across the organization.

The Agile Manager thread also has negative indications of some of the three mechanisms. Cathy’s local needs, for example, have made it advantageous not to incorporate Agile Manager into her practice. Delegating that work to Jim effectively blocks mediation. She is unlikely to learn the tool the same way as he does, and that deliberate absence of mediation reinforces a division of labor.

Moving on to the other threads of activity will add to the growing map of the user experience team’s collaborative ecology, and at the same time reveal more examples and insights about the teams’ selection, practice, and mediation of that ecology.

**Thread 2: Collecting User Data**

User research is the most important “lever” that the user experience team can use in the software development lifecycle (Reggie). Many of the team members I met with, including Reggie, Jim, Cathy, Sean, Steve, and Tom, all mentioned it as part of their work. User research permeates the SDLC, informing project planning and strategic decisions, verifying and modifying the direction of Agile drives and sprints, and identifying defects and
requirements for future iterations. Yet while user feedback interfaces with the SDLC at several points, much of the user feedback effort exists outside of the formal “release cycle.” It is not explicitly tracked inside Agile Manager, and many of the activities related to gathering user data are narrowly confined to the user experience design teams—an even smaller niche than designers in general, and so they rely on a fleet of open and informal tools that they use in various local and innovative ways.

The user experience team employs a variety of user research tools and practices to collect, organize, and articulate user data. Around the release of a new software product, the user experience team requests feedback directly from the field in the form of alpha, beta, early release, and six month feedback solicitations (Reggie). On rare occasion the user experience team uses ethnographic site visits to gather data (Sean). Most often, user research is carried out in the context of a customer feedback group, which provides survey and focus group data on a monthly basis (Jim; Cathy). Each of these three data sources generates vast stores of data, but the activity itself remains a specialized pursuit of the user experience team. Since this work is largely separate from the SDLC manager, the team constructs a collaborative ecology specifically to support 1) direct feedback from customers, 2) ethnographic/contextual inquiry, and 3) customer feedback group.

The least-discussed tool the user experience team uses to collect user data is direct feedback. This is often an informal collection of insight and opinion. In one form, the user is provided with a product, either in alpha or beta form, in an early release program, or as part of a normal release plus 6 months of experience (Reggie). The feedback professionals then simply ask for the users’ impressions without setting up any formal study. Other times, the user feedback specialists specify some questions, and ask customers to keep a “diary” of their
experience with live code (Cathy). Cathy relied on her personal relationship with customers, enacted through communication tools like email and telephone calls, to conduct this kind of study.

Besides simply asking customers to share their perception of a product, the team has also planned formal usability tests. These typically occur on-site with the user experience department. On-site tests occur on a regular schedule a few times a year, with customers situated in a full-scale usability lab. Cathy did mention that, despite the regular schedule, on-site testing has become less common in recent years. It has occasionally been replaced by virtual tests, where the customer signs in to the test machine remotely. Regardless of whether the test occurs on site or remotely, Cathy and Jim tend to record video, audio, screen-casts, and typewritten notes during the test (Cathy).

This kind of direct feedback seems to be held over from a previous set of practices. As Cathy explained it, on site testing and direct feedback, while currently infrequent, had historically been a key part of the user experience team’s process. Onsite ethnography and direct testing seem to be an ideal held in accordance with a previous way of doing usability studies. Those practices remain important, they still generate influential user data, and are still esteemed by the team, but for unclear reasons they have become relatively less common.

If, as I suspect, direct usability testing was the paragon of an older model of user experience, the current idealized paragon is likely ethnographic inquiry. Ethnographic site visits also contribute to the data-gathering efforts of user experience professionals, though far too rarely in the mind of the user experience team (Sean). Ethnography is an intellectual tool in its own right, shaping what the user experience team does when they collect feedback and user data. Its goal is to “reach a shared understanding with the user how they do their tasks,”
(Sean). But it is more ideal than actual. In fact, while it was mentioned occasionally by user experience team members (Sean), and on several occasions during my time on site, I only observed it used on one particularly large and well-funded project (Sean). Ethnography exists in the collaborative ecology as an intellectual tool, but also as a powerful symbolic entity used to shape the character and representation of the user experience team. Even on the occasions it is really practiced, it was an abbreviated version of what the user experience team would really prefer.

In the one ethnographic project that Sean explained, the team developed a formal plan to guide the team’s tool use or general methodology, in the form of a “practitioner’s handbook,” developed specifically for the project. In the face of time, funding, and participant availability began to shrink, the team abandoned their plan and “ended up just kind of winging it,” (Sean). Switching to a more liberal strategy seems to have had a side effect of allowing much more variability in the selection of tools and practices used by the team members.

Sean’s ethnographic project identified several tools that supported his team’s data collection and management. Rather than using recordings or transcriptions they relied on “a notebook and a pen…or the electronic equivalent which is WordPad or NotePad,” (Sean). Sean himself preferred an in-house tool designed specifically for recording events in usability tests that I will call Usability Notes. In essence, it is a text editor that inserts a timestamp for each line written. Sean’s use of Usability Note seems to be a bit of an anomaly among his colleagues, and is indicative of Sean’s character. He is an early adopter who develops a deep commitment to his preferred applications, building expert knowledge in relatively niche
tools. Having expert knowledge of several informal and open tools gives him an ability to improvise and repurpose even outdated software and use it for powerful new purposes.

Sean’s preference for Usability Notes is an example of his particular ability to improvise and repurpose tools. He uses it “for a lot of things other than just logging usability tests,” because it starts with a time-stamped entry and allows him to add codes to the notes (Sean). From there, he can take the output and manipulate it to create Excel Spreadsheets (Sean). Then, he can use these spreadsheets to do further manipulations, to add categorization to individual notes, and to export it for other uses as well. In his words, it is, “extensible.” Using Usability Note at all for interviewing is in itself a repurposing of the original tool’s design. It was initially intended to record events during usability tests, not during ethnographic interviews, and there is little indication that it is widely used. Sean was the only person to mention it, and his use seems to be an anomaly that doesn’t spread beyond him, regardless of the benefits he himself gains.

Based on the reports of participants and observations while on site, ethnography is not a common practice, but it is considered an ideal. On the occasions when it is used, it produces vast amounts of note data that is likely difficult to organize and use, owing in part to the contingent and variable practices for collecting and recording data.

By far the most consistent and productive form of user research is a monthly teleconference that Jim and Cathy host with their customer feedback pool, some of which I observed while on-site. The teleconference call acts as a virtual focus group where customers volunteer their time in order to influence designs during development. Jim and Cathy prepare the calls in response to requests from project leaders. Organizing around particular projects and topics leads the teleconference to generate feedback on a narrow set of issues.
Teleconference often consist of a user experience designer doing walkthroughs and demonstrations with early prototypes, while the hosts ask questions, do surveys, polls, and all manner of discussions. This helps the team “get a feel” for the customer’s reaction, (Reggie).

Every monthly customer call uses audio recording and screen-casting tools to create a record of the conversation, but the hosts seldom go back and review the recordings, relying instead on typewritten notes kept in MS Word. Cathy attributes this practice to the team’s lack of a tool that would make all that information easy to use later on. She says,

Well even our recording tool is a shareware, we can't buy software, so Camtasia would do all that for us...We can't get Camtasia, and if we did I’d have to try to learn it, which would take time and all that, but exactly, there's nothing easy. (Cathy)

She would like to take live audio or video recording of users struggling and use that for presenting feedback later on. As Cathy says, “It’s best if you can actually see the problem, the kind of live stumbling and stuttering kind of thing,” because that sort of data has “more teeth,” but in the absence of a tool that makes it easy (Camtasia) they choose not to. Their “poor substitute” is to use quotes from their typed notes. Cathy and Jim have a constrained selection to make regarding the tools they use, and as a result they are not able to engage in what they report would be a preferred practice.

Cathy and Jim use polls and surveys to gather data during teleconferences with customers. Aside from the obvious benefit that it’s an “easy way to get hard data” (Cathy), it also serves a secondary purpose of guiding the focus of group discussion. According to Cathy, “a well-written poll question brings (the discussion) back on the same page,” (Cathy). Consequently, Cathy and Jim have developed an emergent practice of interspersing normal discussion with quantitative data gather techniques; even if the polls and surveys don’t produce a lot of insight themselves, they still might prompt useful comments and discussion.
This practice seems to have emerged naturally from the exigencies of the user feedback process.

While the Jim and Cathy facilitate discussion using polls and surveys to help focus the conversation, they also sometimes maintain an open text chat in order to provide a conversation space hidden from customers. The chat is populated by the developers, designers, and other stakeholders from within the organization. The chat gives an opportunity for a linguistic backstage, where the persons who requested the topic of the feedback session can guide the user experience professionals running the sessions by answering questions and asking for further inquiry on some topics (Cathy). As a tool, the running text chat provides a layer of coordination behind the scenes of the data-gathering effort. It is an entirely untrained and emergent practice that re-enacts the “mirrored window” of traditional usability labs in a virtual context.

Because gathering user data is highly specialized activity that is essentially unique to the user experience teams, it is insulated from broad institutional practices and conventions. Despite the existence of a corporate user experience group, there is no rigid standard way of doing things, and so the individual user experience team members develop their own varied technology practices. This leads to innovation, but it also leads to messiness and improvisation when it comes time to sanitize and present user studies within the framework of the software development lifecycle.

The user experience effort to collect user data through direct feedback, ethnographic inquiry, and the monthly customer feedback pool produces a wealth of data to curate and use, consisting of customer comments, audio recordings, screen-casts, typewritten notes, survey results, emails, complaints, and other artifacts. The stores of user data are massive and
unwieldy, and in their raw form largely unusable for any sort of persuasive effort to drive user-centered improvements into the product. In order to make these data useful for reinforcing new requirements, guiding product development, and informing design, they have to be packaged into a consumable form.

The user experience team takes user data and packages it into reports and presentations. By packaging data, the team “starts from the customer comments, or more likely a distillation of a number of customer comments, into a single pain point or use case that customers feel they need,” and ends with a powerful lever for influencing the development lifecycle (Reggie; Sean). Packaging data accurately and copiously is important because “The better our data in a presentation is, the more likely they’ll listen and the more likely we can get something changed,” (Cathy). It is also common for managers and stakeholders to challenge conclusions from the user experience team, forcing them to trace their conclusions and pain points to raw user data (Sean; Cathy). Consequently, the team has a standing need to express user data in a concise and persuasive way, distribute that expression to the development team that needs it, and also to trace the expression back to the data that originally produced it.

Once the user experience team has a set of conclusions expressed as a report, they have to distribute it to the appropriate members of the project team. Historically, reports have been stored in a shared proprietary database which I will call Team DB. Sean is the main promoter of this tool for coordinating teams and storing documents. In his words,

I tease other people and they tease me about our preferences for things like that, you know, when we say, we need a document repository somebody, maybe tends to be a younger person on the team says, well ok, we'll create a wiki, and I'll say how about we create a [Team DB] and people roll their eyes and you know. hehehe. So there are definite differences in how we approach those questions. (Sean)
Sean is so closely associated to the Team DB system that he tends to be the person who administrates most of them for the user experience team and who creates new ones when they are called for, even when he himself is not one of the collaborators involved. This special association makes him particularly influential when it comes to storing and sharing files.

While Team DB is a well-known tool for storing and sharing files like user studies, it has some drawbacks. First, like Version Control Suite, it is being phased out in favor of a newer web based technology that I will call Connector. Second, Sean is one of the few people who are truly expert with Team DB, further cementing his role in administrating it (Jim). Third, functionally, Team DB has strict accessibility rules. As Cathy states,

Before that we would put our reports in our HCI [Team DB], so just the usability people could get it then and we would send it out in email, so you had to send the file to everybody on your distribution list. (Cathy)

As long as reports were stored in that Team DB, Cathy had to maintain an email distribution list and manually circulate reports to whoever needed access—especially stakeholders outside the user experience team. So, using Team DB caused the user experience team to develop a variety of practices to compensate for the difficulties in administering it and for its strict accessibility rules.

Despite its quirks, Team DB has worked well for the user experience team, but it is being phased out in favor of a web based groupware product called Connector. As Sean states, “they're trying to get us not to create any more [Team DBs] or tools of any kind and to go instead to this [Connector] product which is a web based product.” (Sean 9-128). Connector includes blogs and forums and wikis and file sharing. While the user experience team and many other small teams continue to use Team DB, the overall product team has had
a Connector running since the tool first became available. It has been gaining users on a project-by-project basis. Reggie, for example, created and promoted a corner of it for a project on measuring user experience quality (Cathy). Eventually, Cathy suggested to Jim that they attached a page to Reggie’s wiki for user experience reports. From my interviews with Cathy and Jim, it seemed like the initial reason for selecting the Connector wiki was mostly experimental, but they have come to prefer it because it solves their accessibility problem. They no longer have to maintain distribution lists for reports of user studies. Consequently, the practice of using the Connector stabilized into a consistent convention for reports of user studies.

Of course, the Connector tool has drawbacks of its own, largely related to the wiki system’s usability. Almost universally, everyone in the study dislikes the wikis. Steve, perhaps, says it most poignantly. “I probably should put my (designs) out there because [the Connector Wiki] is the project repository. I just never think to do it because I hate those damn things,” (Steve). Paradoxically, the team of usability experts all seem to agree that the wikis are awful to use, or at least that they have “a lot of overhead” (Reggie), and yet many of them (Carol, Jim, Alice, Kris, Reggie, Steve, …) seem to express feelings of guilt about not wanting to use them, even among the members like Reggie and Cathy who promote wikis. Of the team members I interviewed, Reggie is main wiki promoter and he seems convinced of their usefulness as a collaboration tool despite its usability issues, but the rest of the team seems unconvinced that the benefit outweighs the cost of learning a new system, especially since they can still rely on Sean to handle a lot of the overhead of things like the Team DB system.
In regards to storing and sharing usability reports, accessibility seems to outweigh usability, and so the Connector solution wins out. Consequently, user study reports now exist as word documents attached to wiki pages inside a massive social collaboration suite, developed by the same internal team that developed Team DB and the corporate email system. The process of phasing out the older Team DB system continues, though, and the practices for publishing documents vary from team to team and project to project.

While the user experience team takes pains to make reports of user studies available to broad groups of people, the raw user data is often left in a network drive known to only a few people. And that’s an improvement; Cathy took years to convince Jim to unload the data from his local hard drive (Cathy), so it would be backed up nightly and also possibly to make it accessible to others. Jim’s and Cathy’s memory are the only index of this old user data. Because the raw user data goes into a sort of black hole, it is difficult to build longitudinal assessments of usability improvements and to incorporate historic data into current conversations. The team sees this as a weakness in their collaborative ecology.

The user experience team, according to Cathy, has a memory problem; their tool ecology lacks an adequate means to curate years of user data. As she says,

You’ve got a tool for showing customers material, whether that’s a PowerPoint that has a mock up in it or a live code situation, and then you’ve got another tool for recording it, and then it almost seems like there’s missing a really good tool to take that kind of data and make it useful later.

That to me is a lack in how to get in to that collateral that we’ve collected for years and years. And it’s just my memory now which is failing. (Cathy)

I asked Cathy to speculate on a tool that could solve her problem. She described a tool to take “raw notes, reports, whatever,” that would automatically categorize the data and make it accessible by searching keywords like “automation,” (Cathy). The fact that Cathy could
articulate her perception of the gap in the user experience collaborative ecology and specifically identify the kind of tool that could fill it is stunning.

What is more stunning is that she seemed unaware that Sean, working in a different office with a different group, had already identified the same problem and improvised a tool to address it. In order to organize his own reams of time-stamped notes from his ethnographic work, he leveraged his expert knowledge of the flexible, if dated, Team DB tool to build a searchable data clearinghouse. In essence, it is like a highly customized version of the standard Team DB, set up with internal categorizations that work like storage directories. He described a typical use of his system as part of a presentation to executives, saying

And we showed them some requirements and if the executive were to ask, well how do you know that's a requirement, or who said that or what exactly did they say, I was kind of poised with this database and ready to be able to look up and say oh don’t forget [Customer A] says that or [Customer B] says that. (Sean)

Because it included all his data, stored, categorized, index, and searchable, he could use it as an advanced research tool, but also as a just-in-time way to retrieve salient user data and reinforce conclusions during presentations of user studies.

Sean’s improvised analysis tool repurposes a general, open tool and may fill the gap Cathy identified, but the tool hasn’t spread beyond his own personal use. So far, Sean has not successfully mediated the tool even to his own team. He has mentioned it in team meetings, and has informed his coworkers that it is available for use, and they have expressed interest (Sean). However, no one in the organization can match his expertise with the Team DB system. Furthermore, no one else really promotes the Team DB system as much as he does, so it may be that the new niche application is held back by its dated platform. Consequently,
while his database tool has become an emergent practice for him, it has not mediated to the other members of his own team despite his efforts to promote it.

Sean mentioned a variety of formal mediation activities to promote his data analysis tool, including demonstrations, instructions, and in-person walkthroughs. He seemed confident that given the chance to demonstrate the tool, write instructions for setting it up and using it, and assist his coworkers setting it up, that it would see more use, as he writes,

I really need to be able to do that demo and help people get their personal copy set up to, you know, really make it, to really get them excited about it and I think they will use it as one of the tools. (Sean)

His faith in demonstrations aligned with the general understanding that live presentation is best tool that the user experience team has at its disposal for making persuasive arguments; (Cathy). But, from an outsider’s perspective it seems like Sean’s opportunity has passed, especially when he goes on to say,

You know these people are busy and we’re kind of at a point where we’re not as focused on figuring out who said what, we’re kind of moving on into designing new things but it’s still always useful. (Sean)

The tool is consequently less relevant to his team’s current activity in the software development life cycle. Mediating his new practice to others may be impossible until a similarly high profile project begins again.

Even though Sean’s team no longer has an urgent need for his tool, Cathy and Jim could benefit from it if they only knew it existed. As previously mentioned, their user feedback sessions occur every month, regardless of the status of any particular software development life cycle. In this case it seems like the mediation problem is about awareness. Sean, as mentioned before, works in a different office with a different group, and his own group is passed the point in their current project where his user data tool is most useful. It
may simply not occur to him to pass it along. Cathy and Jim, for their part compensate for the problem in other ways, relying on memory or simply avoiding the issue. Their practice is stabilized enough that their desire for a better tool doesn’t go past wishful thinking.

While examining how the user experience team gathers, stores, and uses customer feedback data, a variety of new tools have entered our map of the collaborative ecology (Figure 3 next page). They include tools like email, telephones, and teleconferences for maintaining relationships with customer volunteers. They include tools for recording data in video and audio, in screen-casts, on paper, and in note-taking tools like WordPad, NotePad, and Usability Notes. They include real time chat for maintaining a backstage conversation among the development team during user studies, and polling tools for generating numerical data and keeping conversations on track. They include storage and file sharing solutions like Team DB and Connectors, and improvised analytical tools like Sean’s special Team DB.

Figure 3: Tools supporting data collection and customer relationships. The user experience team incorporated more communication tools to maintain a relationship with their customers, and also added a variety of tools for collecting, storing, and analyzing user data.
The user data thread has also spawned new insights about selection, practice and mediation, some of which are worth emphasizing here. For example, the selection of free and open source recording tools shows that instance selections are constrained by categorical selections; Cathy cannot get access to Camtasia, so she uses an open source tool instead even though she considers it inferior. A repeated dilemma arises around groupware, with teams making various choices about using Team DB or Connector to share files like reports of user studies. Practices like on-site user testing that stabilized under older models of usability and user experience remain influential, and continue to be productive and to guide practice, despite increasing difficulties in employing them. Other practice, like the tools and strategies used to conduct an ethnographic study, seem to not have had a chance to stabilize at all, resulting in a wider variation of tool choices and practices.

We also saw deliberation and compromise in the selection of tools, particularly in regards to the Team DB versus Connector dilemma. Despite the prevalent distaste for wikis, the team remains willing to use them, but not exclusively. As long as the Team DB system is supported, many teams will continue to choose it and Sean will continue his informal role as Team DB administrator. Cathy’s and Jim’s particular need to make reports of user studies available to a wide community reinforce their continuing decision to make use of a wiki that they don’t really like.

By far, though, the most compelling insight here is about mediation. As seen in Sean’s user data analysis and curation tool, timing and opportunity is a key requirement for a tool’s use to spread through an organization, whether formally or informally. So long as his demonstrations and instructions are aimed at a group of people with no immediate need for his tool, his efforts are not likely to be successful, even while his tool seems quite relevant to
another group of people. For their part, they have routinized their strategies for compensating
for what they see as a deficiency in their collaborative ecology, and don’t expect or even seek
a solution.

**Thread 3: Supporting User Requirements**

The user experience professionals I worked with tend to consider advocating for the
users to be their primary role, and to fulfill that role they take an active approach to
generating software requirements. Software requirements are expressions of desired
functions and attributes to be realized in the course of a project. Stakeholders from different
departments, like architecture and marketing, can contribute software requirements (Ray;
Steve; Tom; Reggie; Sean). The user experience team, on the other hand, is most directly
responsible for user requirements. For them, *users* are supposed to be the source of
requirements, and indeed the user experience professionals spend a great deal of effort
interfacing with users, collecting data, and “distilling” or “aggregating” that data into user
requirements (Sean; Cathy; Reggie). Requirements supported by user data (Cathy) become
influential artifacts of the overall software development life cycle, and a distinct part of the
collaborative ecology supports the process of identifying and prioritizing them.

The user experience team members, especially the designers, have four
responsibilities in regards to producing software requirements. First, user experience
designers draw on user studies to determine requirements. Second, they translate user data, or
system and marketing requirements, into user-centered “stories.” Third, they insert their
requirements into the Agile Manager Priority Tool. Finally they advocate and promote user
requirements to help ensure they “make it in to plan.” The next few pages explain these four responsibilities.

User experience designers plan and execute user studies to elicit requirements. Designers need to be concerned with all requirements, regardless of whether they originated with systems architects, marketing, or some other department (Ray 38, Tom 32). But they are also responsible for injecting requirements of their own based on user feedback (Reggie), and so they carry on their own specialized activities for generating and articulating requirements by interacting with users. Taking advantage of the data collection efforts of direct feedback, ethnography, and the monthly feedback group, the user experience team takes user data and packages it to be more persuasive with decision makers (Cathy). The packaging activity results in a profile that serves as an imagined typical user. As Sean said, “We rarely develop with actual users, more of an aggregation.” User data, or better, data from a handful of users (Reggie), is seen as more powerful data that is useful for moving user centered requirements through the priority system and subsequently through the software development lifecycle manager. The software tools that support this process of eliciting requirements is largely the same as that supporting the collection of user data already described.

User experience designers translate user data and system and marketing requirements, into user-centered “stories.” Once the user experience designers identify requirements either from their own “pretty good feel” for the users’ perspective (Reggie) or from requirements proposed by other departments (Ray; Tom), they translate the requirements into user-centered goals. In Ray’s words, requirements are often initially expressed as “system requirements and we have to maybe backtrack a bit and get our mindset right” (Ray). When system-centered requirements come from other departments the user experience designers
“reverse engineer what the high-level point of a marketing requirement or an architecture requirement is” (Ray) to “piece together how a user would benefit,” (Ray) to “understand and flesh that out more in terms of user goals (Ray). The practice of expressing requirements from a user’s perspective is consistent across the whole user experience team.

The consistency in expressing requirements from a user’s perspective comes partially from the team’s self-proclaimed identity as user advocates and partially from an institutional commitment to Agile software development. Agile uses narrative metaphors to express requirements from a user point of view. So, after collecting requirements through user data, and translating other stakeholders’ requirements into user-centered goals, those requirements are expressed as “stories” and “epics.” Stories are short, formulaic expressions of functionality from a user’s perspective. For example, “As an end user, I need to be able to sort my email by date received.” Related stories are collected together into lists called “epics.” Epics and stories are then organized into sprints. Sprints are short cycles of development during which an entire set of stories is completely developed from start to finish. Sprints, in turn, are carved into tasks and work items that can be assigned to individual team members. User stories, therefore, are an extremely stable micro-genre developed for Agile software development that has implications all the way through the software development lifecycle.

Once requirements are developed into stories, another responsibility of the user experience team is to push requirements into the formal software development lifecycle. Because the overall project teams’ resources are always limited, only some requirements actually “make it in to plan,” (Cathy) and get developed as software. Reggie, as a “user experience lead,” is particularly responsible for leveraging user data and user study reports to
promote user-centered requirements. This activity helps guarantee that the user experience team has a positive effect on the end product. The first step is to insert the new requirement into the Agile Manager Priority Tool.

The Agile Manager Priority Tool is a web-based application for managing requirements and forming them into a project plan. The tool acts as a central place where all the requirements from various departments are brought together and prioritized (Reggie). It is integrated with Agile Manager, but not strictly a part of it. While Agile Manager is an all-encompassing tool accessed by almost everyone involved with a project (Jim, Alice, Ray, Tom, Reggie, Sean), the priority system is used only a few of the user experience professionals. That group includes Jim, Sean, Reggie, and presumably some of the other designers. But Reggie explained that it collects requirements from all over the organization, including user experience and seventeen other sources. Once user stories are in the priority tool, the overall development team chooses a set of them to complete end to end (Sean).

Reggie in particular is the user experience team member responsible for advocating and promoting user requirements to help ensure they “make it in to plan.” Once a requirement is entered in the priority tool, there is no guarantee that it will be placed in a sprint and assigned as a task to a developer. The challenge is explained by Cathy:

So we opened up a requirement went in to the tool, [Jim] did it but we were all sitting there, and typed in all the fields … and pressed enter and now it’s a requirement. Well you know in the scheme of the other 200 requirements…that want to get in to the next plan, who's going to get that through? It just falls out. If somebody with power pushes it, whatever, I don’t know, but of course it didn't get in to plan yet. (Cathy)

Requirements can float in the priority system indefinitely (Cathy), so Reggie uses emails, instant messages, and presentations, along with samples of user data (Reggie), to reinforce and emphasize the importance of user-based requirements. The interesting thing about the
role of the requirements tool is that the tool itself is not enough to handle the job. In fact, a large part of Reggie’s work is to manipulate the normal operation of the requirements tool by advocating for user-centered requirements and tracking them to measure the effectiveness of the user experience team.

That advocating continues even after the user stories are entered into Agile Manager as sprints. Even though a requirement has made it into plan and its expression is fixed, the interpretation of the requirement in Agile Manager remains individual and subjective. This is why so many participants mentioned that it is important to verify that everyone is interpreting things the same (Steve; Reggie; Alice; Ray). The user experience professionals used extended rounds of review and live teleconferences (Ray; Reggie) to help keep everyone on the same page. Furthermore, the user experience team needs to keep track of the raw data that produced requirements, in case they have to answer a challenge. As Sean explained:

We just got through with a round of executive presentations…and we showed them some requirements and if the executive were to ask, well how do you know that's a requirement, or who said that or what exactly did they say. (Sean)

Keeping track of user data and how it connects to requirements is vital because that data is where the user experience team ultimately derives its authority. Effective use of customer data is a key strategy of the user design team for advocating a new requirement or for most efforts to drive a product change.

Aside from requirements, user experience professionals also identify and catalog software defects, or bugs. Like requirements, defects are planned product changes, as Cathy states:

A requirement is something new that needs to be put into a product or taken out or whatever. Defects are you already have something that doesn’t work very well. (Cathy)
It seems anyone can enter them into the system, but often they come from the testing team, or from the user feedback specialists, who incorporate some user data into the defect, often as a paraphrase. As Cathy explains,

I don’t have to but I do, usually….I just make sure to say, I was doing a customer walkthrough and this came up then they knew I wasn't just willy nilly want something changed, I gave them the reason I want something changed. (Cathy)

Defects are not prioritized by the requirements tool; they go directly into the Agile Manager or Version Control Suite and are assigned to whatever developer is responsible for the feature in question. This seems to be because defects represent flaws in already delivered software. There is no question as to the importance of fixing them, only to the priority.

Like collecting user data, identifying and promoting user requirements is a very specialized activity that has a large effect on the software development life cycle. And while fewer members of the user experience team are directly involved with setting requirements, the general activity of determining and prioritizing reveals another subdivision of the overall collaborative ecology with its own supporting tools. The Agile Manager Priority Tool is an obvious example, as is the defect tool. But the tools that really make an appearance here are the informal and general-purpose tools used to clarify and advocate requirements as they move through the prioritizing process and then through the software development lifecycle (Figure 4 next page). These tools—email, instant messaging, telephones, teleconferences, presentations, and the like—are avenues for the user experience team to clarify and add gravity to the changes they hope to see in the product.
Figure 4: Tools supporting the identification and prioritization of requirements. The user experience team added a few more communication tools, but most significant addition to the user experience ecology was the priority tool for deciding what new requirements will become software features.

The requirements process has somewhat less to say about selection, practice, and mediation of collaborative ecologies than the previous sections. It includes only a handful of tools, and they generally don’t have viable competitors. That suggests that the priority tool and the practices surrounding it are largely settled and central to the collaborative ecology. The stability may be a result of a lack of alternatives, or a strong match between activities and tools, or overwhelming meditational effort in the past.

What the user requirements effort does illustrate is that tools that make up this collaborative ecology are not working in isolation of one another—the requirements tool interacts with communication tools and storage tools and data analysis tools in significant ways. The collaborative ecology is extensively interdependent. The user requirements thread also begins to illustrate that customer feedback and planned designs are flowing through, or
being driven through that collaborative ecology, finding expression in different tools as they progress through the software development lifecycle.

**Thread 4: Supporting High-Level Designs and Specifications**

Aside from gathering user data and identifying and promoting user requirements, the user experience team is active in developing high-level designs and specifications. Specifications play an important role in the software development life cycle as authoritative expressions of a plan for a product. As described by various participants (Ray; Reggie; Sean), they typically result from a broad effort in collaborative writing and editing, to which the user experience team adds expertise in interface, interaction, and graphic design in addition to representing the customer’s perspective on things like performance, security, and system attributes. Specifications are virtually always published as MS Word documents.

Specifications develop in parallel with the other user experience deliverables. They usually incorporate preliminary visual artifacts from prototypes and artwork alongside advanced textual artifacts like user stories and feedback data. When they are complete, specifications reside in the Agile Manager along with prototypes, code, tasks, timelines, and other project management information. However, despite the near uniformity of the end stage of the specification’s useful life, the means of production and collaboration leading up to that end stage are local, variable, and contingent.

From the perspective of user experience designers, specification documents progress through three stages that I will call collection, review, and consumption (Figure 5 next page). They begin as a collection of user requirements, user stories, and low-fidelity prototypes that are placed together with requirements and designs from other departments. After creating a
section of the document based on one or more meetings with the broader planning team, the
designer and other authors then circulate drafts for review (Reggie; Steve). This phase
reconciles misconceptions between team members (Ray). A possible symptom of distance
collaboration is a heightened awareness that sometimes stakeholders come to agreement on a
decision only to discover later that each party had a different interpretation of that decision
(Alice). After an indefinite number of drafts, the document is published to the project
management tool to be consumed as the authoritative representation of the product’s plan.

Even though the role of specifications in the software development lifecycle is
relatively stable, the practices and tools that produce the specifications are variable and local.
Agile Manager, which enshrines the patterns of use around specifications, was developed
with Agile software development in mind (Sean; Tom), and it has facilities for all kinds of
activities—tracking changes to code, storing code, tracking tasks, storing miscellaneous
documents like specifications and prototypes, and managing comments. Custom plug-ins
make it work with a variety of modern software integrated development environments
(IDEs), so it can be intimately connected to the process of developing computer code.
However, Agile Manager has little facility for collaborative writing and editing, leaving the team to improvise its own methods.

Since the Agile Management tool has little formal support for collaborative writing and editing, teams “spelunk down” into a “whole fleet of other social tools,” (Reggie). This “fleet” almost universally revolves around MS Word, because “whoever is responsible for the document picks what they know at that point, what they know best” (Reggie), but also includes tools and practices for commenting, sharing, circulating, and merging documents that are “at the whim of whoever the project manager is, how formal they want to make things,” (Steve). Consequently, the individual project manager has increased influence over the eventual collaboration practices that emerge.

MS Word is the preferred word processing solution, but the user experience team members have tried other tools. Word is familiar to everyone I encountered, and its file format is universally consumable. Other word processing tools are available, including an in-house alternative developed by the one of the corporation’s software groups. Ray did try it once, to avoid requesting a new license for MS Office when he received a new laptop, but eventually abandoned it. For him, the internal solution wasn’t sufficiently compatible with his vast collection of legacy designs and specifications, all in MS Word and MS PowerPoint. Ray says, “I hate to say it but one key thing to picking a tool is, is it stable?” Participants were also aware of services like Google Docs, MS Fileshare, and MS Office 365, but those were not considered serious options, likely for security reasons (Reggie). So, on account of license costs, security concerns, universal consumability, and individual familiarity, the organization prefers the MS Office productivity suite.
I did encounter one exception, when Reggie led a collaborative editing task that used a wiki as the word processing tool. He used a Connector wiki as the “the actual content editor” saying:

Yeah, and I've chosen a couple of different ways. Earlier I would choose a common wiki and that was the actual content editor, it worked ok, there is a lot of overhead in a wiki if you need to add images and all that stuff at least with the versions we used at the time. With some of the latest html 5 wikis editing photos is a bit easier…

But at the time it was a lot of work to upload the image and actually embed it into the wiki, it was a multi step process, rather than a word document where you can just drag it in there and it just formats automatically….

But generally I would use... Word document as the actual editing tool and then a shared wiki or a shared site to have master version that people can edit from and then I would do the merging. (Reggie)

This exchange suggests that Reggie is a trendsetter in the use of Web 2.0 tools, willing to experiment with new ways of doing things even if this particular practice has not previously emerged as a viable convention. By his own analysis, upcoming advances in functionality based on HTML5 may make wikis more viable ways of shared writing and editing within this collaborative ecology, but it remains hard to imagine it replacing the MS Office Suite for this team.

Despite a handful of experiments in other tools, it is clear that MS Word is deeply ingrained in the practices of shared writing. The in-house tool is quickly abandoned for productivity concerns, and even the experimental wiki method eventually resulted in a Word document to make it fit into the conventional expectations of the software development lifecycle:

There certainly have been times that we've edited a wiki as a spec and then at the very end we just said export as word doc so we could quote publish it. (Reggie)
Furthermore, local practices of collaborative editing with tools like MS Word’s tracked changes are deeply rooted in the organization’s way of writing, even as variable as it is. To understand this juxtaposition of variation and standardization, let’s examine the practices of team production.

Whoever is in charge of a specification effort has a lot of influence not only on the collaborative writing tools to use, but also on the emerging practices for using that tool. The person in charge acts as coordinator, editor, and final decision maker for producing the specification, making lots of decisions on the team’s behalf based on what they are comfortable with. One of the first decisions the editor makes is whether to facilitate editing serially or in parallel—do all collaborators make changes simultaneously, sometimes causing the editor days of effort to merge the changes? Or, do collaborators make changes one at a time?

Teams can edit documents in parallel, producing multiple simultaneous versions that have to be integrated together. The general editor typically schedules periodic conference calls to handle disagreements and controversies. Then, the text of the specification is circulated through, often with email but taking advantage of other storage tools as well. Each round of editing produces multiple drafts from the different team members, whose identity is noted within the document or with a code affixed to the filename (i.e., “Spec001-ux.docx”) (Reggie). At that point, the editor uses the merge function (Ray; Reggie) of MS Word to create a new version, and spends time—sometimes days—incorporating the changes and trying to satisfy every contributor without losing the overall unity of the document and giving the document a similar voice (Reggie). Finally, the editor re-circulates the document so collaborators can verify that their changes are accurately represented. The parallel edition
process can repeat many times before any given document satisfies all the collaborators (Ray).

Failing to establish a common practice for collaborative editing can cause an extraordinary amount of overhead work. An example is the conventions for using tracked changes. Tracking changes is a familiar means of identifying who modifies a text and how, similar to revision histories in wikis (Reggie). MS Word’s tracked changes were mentioned commonly in interviews (Ray; Reggie; Sean), but as Reggie explains varying practices for using tracked changes can cause difficulties. These difficulties were most clearly illustrated in a case already mentioned, where the editor took it on himself to incorporate all changes from all other contributors. Reggie pointed out that contributors who used inline tracked changes exclusively were the quickest to evaluate and integrate, while contributions made with comments or inline notes required extra scrutiny and effort. To mitigate discrepancies in how tracked changes and other collaborative editing tools are used, Reggie reported that one manager began a project by explaining how tracked changes should be used, but issued no corrective statements to the team or to Reggie himself. This effort to guide the practical use of tracked changes is an example of an informal attempt to mediate a preferred tool use.

Other teams and managers, particularly in Sean’s group, prefer a serial approach to collaborative writing. Serial editing has the advantage of avoiding some of the hassle of merging multiple versions in to one. It can be easier to identify the current authoritative version, since only one version circulates at a time. Sean says:

We do sometimes edit kind of jointly, we use tracked changes to try to coordinate so if I’m making changes to ‘A’ then I kind of get the virtual pen and nobody else is supposed to be editing, and you know I’ll save my changes and hand it off to someone else because otherwise you know tracked changes and combining documents, merging documents doesn’t always work that smoothly. (Sean)
As each collaborator completes a revision, she merely advises whoever has it next that is available. Sean’s team does this process in serial, because in his words merging is not always “smooth.” His team has developed an alternative conventional practice to address the difficulties associated with merging a document.

Serial editing brings on its own coordinating challenges, like identifying who is supposed to have “the virtual pen.” Agile Manager handles this problem for software code by implementing version control and locking files for editing, but the feature doesn’t seem to be used for other documents. When Reggie faced this problem he improvised a tool based on his experimental groupware of choice, the Connector wiki.

I had a ‘how we're gonna do it’ document or it was actually in the wiki right next to the original file, you know so put your name here. In this case it was, in this case it was actually a little different. It was a common place to put it and we actually did have an informal check in check out, we actually put in red, under the file, "Reggie has this" and when I actually made my changes and put it back in, I would say put it in blue and say nobody has this. It was very manual and it ended up everyone did it and it worked pretty well. (Reggie)

By his own report, the improvised solution was successful, and he only had to correct his teams’ behavior once when a collaborator forgot to mark the file ‘available’ on finishing her revisions. Still, despite his efforts to mediate the practice to his own team, it doesn’t seem to have spread to other projects, either the ones he manages or the ones managed by others.

Over the course of working on a single document, teams sometimes employ many different tools for storing and tracking changes because they want to restrict who can access the document. Sean explained this, saying:

Well we had…a user interface high-level design, which is a Word document…That got published…our local team version of it goes in a [Team DB] so that everyone in our team has access to it and can file an updated copy…so…the document is an attachment to the [TeamDB] and then usually we…layer the most recent version of the document on the top…and so we do sometimes edit kind of jointly. (Sean)
Yeah, that’s more, the [TeamDB] is kind of for the restricted use of our consumability team…and then at some point when we're ready or when the deadline is here, whichever is earlier, we will publish that and for this particular project they're using this odd file repository that's actually hosted over in Germany and I’m sure there's a really long story about why we're using that particular repository but it’s a little bit like…[Version Control Suite]. (Sean)

Sean’s explanation tells the story of a nascent specification that, initially, is tightly controlled and, as it becomes more developed and more authoritative is circulated more broadly and in a more stable location. At an early stage, when the designer wishes to control access to her specifications very tightly, he might only email it as an attachment to one collaborator at a time, or to a small group of colleagues as Sean does with his Team DB. This reflects Jim and Cathy’s use of a wiki to make user study reports accessible to a whole project team (developers, designers, architects, etc). These middle ground locations allow the writer to circulate resources to different teams in a controlled manner. Teams can skip these intermediate storage locations too and place work directly into Agile Manager. Skipping the shared storage step may be more typical for single-author efforts like design prototypes. But even for collaborative editing, the entire “fleet of social tools” Reggie mentioned is an amorphous collection of software assembled as needed. When it is sufficiently complete, the design document finally enters the software development life cycle as a shared document in Agile Manager.

This discussion of the activity surrounding high-level designs and specifications has revealed another corner of the overall collaborative ecology, especially about the tools and practices involved with writing, editing, and distributing collaborative documents (Figure 6 next page). Microsoft’s Word, for reasons of universality, security, stability, compatibility, and familiarity tends to be the central tool, and it is favored over the internally developed tool
as well as a collection of cloud-based alternatives. The moderately successful experiment using the Connector Wiki as a collaborative editing tool was an equally interesting experiment, since it was nominally successful and yet not really repeated. Within the MS Word suite, we saw a variety of emerging practices revolving around tracked changes, with some contributors using comments, some in line changes, and some with general commentary in email and in the document itself. These distributed choices had a huge impact on how the overall editor interacted with the document. Alongside the collaborative writing and editing tools, this thread also began to show the political and organizational importance of storage and access. Team DB and wikis emerged again, used in concert in order to control how documents circulate at different points in their life, and we have just started to see the
role of Agile Manager as a repository of exclusively authoritative expressions of a finalized
design. As already pointed out, that role is true for prototypes as well.

The thread on high-level design and specification illustrates some new conclusions
about selection, mediation, and practice. For example, the ubiquity of MS Word as the
writing tool of choice is interesting, showing such a uniformity of practice and an extensive
mediation that even successful alternatives are short lived despite successful experiments.
Competing paradigms of collaborative editing strategies, like parallel or serial and comments
or in line tracked changes indicate that selections can be extremely stable even while
practices vary widely. Improvised use of general tools, like using the Connector Wiki to
support an improvised check-in/check-out process, show the importance of having flexible
and open software in the “fleet” of supporting tools.

We just begin to observe the politics of storage practices. The team maintains lots of
tools for storing and sharing documents, all of which are capable of the same basic functions.
Yet, despite having the same basic function, different practices develop for each tool
depending on its particular quirks. For example, some have tight access control, others reach
a wider audience. These small differences contribute to surprisingly consistent behaviors in
different teams.

Thread 5: Supporting Design and Graphic Prototypes

A last important thread of activity in the user experience collaborative ecology is
producing designs and prototypes. Visual artifacts are central to the graphic and interface
designers, and to the user experience team in general. To quote Jim, “We are always looking
at something,” For many team members, graphical, interface, and interaction prototypes are a
primary responsibility. Since visual communication is so central to the team’s effort, most members of the user experience team have become experts not only in manipulating the graphical tools that produce visual artifacts (e.g., Adobe Photoshop), but also the tools that control access to those artifacts and circulate them for use.

The visual artifacts of interest come in two forms, prototype interfaces and interactions and artwork for icons and graphics. Both forms are typically produced by a single creator working with a collection of visual tools. The artifacts are rarely edited collaboratively like specifications, but they are often circulated and displayed for feedback from various stakeholder groups. In this section, I explain the production of visual artifacts, their circulation, and their eventual use as authoritative expressions of power.

Interface and interaction designers, as well as graphic artists, use several image editors and visual design tools. Obviously, “translating requirements into visual designs,” as Ray describes his work, requires image processing and some level of drawing function. This includes industry standard tools like Adobe Photoshop (Carol, Steve, Ray, Reggie, Sean). Graphic artists also use more specialized tools like Adobe Illustrator and CorelDraw (Carol). These are powerful tools that take a lot of specialized expertise and as such represent a major learning investment.

The graphic arts community’s continuing use of CorelDraw is a particularly interesting case, since it is both redundant and very old. Graphic artist continue using it because they know everyone worldwide can consume its product (Carol). In particular, it was important that the artwork could be translated. This restriction was clear to the graphic artist on the team, but was far from the minds of the interface designers, who even seemed
unaware that Carol was using CorelDraw. Without the need to print worldwide, only Photoshop entered the interface designers’ minds.

While translation is apparently not an issue for the interface designers, they are certainly concerned with consumability, leading them to adopt MS PowerPoint as a common design and prototyping tool. Since MS Office has virtually saturated the organization, it is nearly guaranteed that anyone in the organization will be able to open, view, and edit PowerPoint documents. As Ray explains:

Everyone knows how to grab a PowerPoint that somebody used for one part of the [product], even if they're doing a different piece, so they can pull out what they need from what’s common and throw out the specific stuff and then add their own specific stuff, so it’s just a commonality of skills sets and who can use what tools also drives the final tool selection I think. (Ray)

He considers it “the lowest common denominator.” Sean even mentioned delivering a conference presentation on using PowerPoint for prototyping (Sean), a message that seems to have stuck with this team.

Ray’s comment that PowerPoint is the lowest common denominator reflects a general dissatisfaction with PowerPoint’s capability as a visual tool. PowerPoint is, after all, a presentation tool and not a design tool. Consequently, designers typically rely on other software like Photoshop and Gimp to do more detail work. Photoshop is widespread (Carol, Steve, Ray, Reggie, Sean), though one designer (Reggie) expressed a preference for GIMP, which is a free an open source image editor. In his words, “for economic benefit the free version GIMP is better than Photoshop. And to be honest they both work pretty well. With GIMP, I've grown to learn its quirks and it’s quite powerful,” (Reggie). But he also admitted being less familiar with Photoshop and not necessarily knowing the difference. From his report, the selection was primarily based on financial concerns.
Despite Reggie’s report, I observed little evidence that the overall institution actually discouraged Photoshop. Photoshop was in fact far more common than GIMP. But obtaining a license for Photoshop or Illustrator or even MS Office would require special requests and justification and money. Selecting Gimp avoided these procedural steps, effectively moving decision-making power from the management class to the individual worker. And, since GIMP and Photoshop were used for production and not communication or consumption, there were no ill effects; I never observed Reggie requiring someone to download GIMP to view his work, he always translated it into a PowerPoint for consumption.

Even though prototypes are rarely produced by more than one person, it is standard practice for designers to circulate prototypes to stakeholder groups to get feedback, so shared storage is again an important tool. The available storage tools are the same as for all the other deliverables—Team DBs, shared network space, Connector Wikis, Agile Manager, and others. Sometimes they are stored locally on the designer’s client machine and emailed to particular collaborators. This is often the case for earlier prototypes and for cases with a small number of reviewers. Larger groups tend to use more the more sophisticated groupware.

Like sharing specifications, choosing a tool to share prototypes is about controlling who will get access, but designers also seemed concerned with how those readers will perceive the document’s authority. As I observed, in different locations, the prototype has a different level of authority. As Tom explains:

I have a weekly meeting with the UI development team so I would send something around as an attachment in email and then… I incorporate comments and then I might put it into [Agile Manager] when it’s sort of been stabilized…it’s not really permanent, it still iterates, but more slowly. (Tom)
In more open spaces, like shared drives, wikis, or in Tom’s case emails, the prototypes is a conversation starter to open to debate on things like “doability” (Sean) or on design in general. In the more formal spaces like Agile Manager, the prototype becomes an authoritative representation of the desired end product. Tom, again, explains:

Yeah, if it’s got things in it that are going to change tomorrow you don’t want to put it out there for people to start developing to. (Tom)

So, for this team different storage tools are useful to limit who sees prototypes at a given stage of development, but also to manage the prototype’s perceived authority and to account for the audience’s expectations. If the design is stored in Agile Manager, the developers assume it is their target.

While Ray’s use of different storage areas for prototypes mimics the storage solutions for specifications, others used different storage solutions to control collaborative relationships with coworkers. This is particularly true of Carol, the graphic designer on the team. As a reminder, she focused on creating artwork for logos, icons, and other graphics both for interfaces and for print and online documents and product packaging (Carol). Like the interface designers, she also circulated her work for feedback, but she was also very conscious that someone else would take her work and incorporated it into a product for release. In fact, she frequently received request for access to a library or repository of off-the-shelf graphics, but she tends to deny the request (Carol) and not provide one, even though she had one available that would be easy to share.

The visual designer in this study had several reasons for not sharing an artwork library. The main reason for her to retain control of access to her artwork was that she (and
the visual designers she spoke for) wanted to promote consistency in how their art was used.

She explains:

We're constantly developing new styles to stay up to date with technology and the look and feel of other products out there, so many times what might be cool ten years ago is not so cool looking anymore and we really don't want people using them, but because they sit out there in that repository people will tend to use it. Then there's the problem of mixing styles. So as we develop a new style for a set of icons and it’s placed in a repository with older styles, but yet you need a metaphor that exists over here but it is fifteen years old, and I need this other metaphor which is this whole different style, but it’s a newer icon, and you mix those in your product you get a really awkward looking interface when you don't have a visual designer to step in and say ok, we've got to update your whole set to have a consistent look and feel. (Carol)

Carol is personally invested in the quality of her artwork and the integrity with which it is used, and she has a need to ensure it works correctly. To hear her, it sounds like a common opinion among the corporation’s graphic designers.

Rather than release her work to the broader institution on a wiki or a Connector page, she used a personal web space and an improvised download site she edits by hand and with Adobe Dreamweaver (Carol). She has maintained this site since the organization first began widespread use of Internet technologies nearly two decades ago (Carol). When there were more visual designers working at the company, she also maintained links to guidelines and visual standards that would be of use to herself and other artists (Carol). Her personal site, therefore, became a sort of information hub. When other visual designers had questions or needed assistance, she could easily forward them a link to her own site as a collaborative shortcut. Even as visual designers have been stretched to the point where it is uncommon to have more than one working on a project, she has maintained this site as her own personal, improvised repository.
Carol has a novel way of mediating her improvised practice, particularly when she transfers responsibility for a project over to another designer. Carol says:

I’ve since turned this over to [Anna] in [China]… because she's taking those fulfillments. But this is a website that I initially created that I post all the labels for all of our products. (Carol)

When a project moves out of her purview, she essentially passes control of her improvised repository to the new person. Whether intentional or not, the activity translates a bit of her collaborative ecology to the new owner’s.

Aside from circulating designs for feedback inside the corporation, designers (Reggie; Sean) and user feedback specialists (Alice; Jim) are also concerned with circulating designs outside of the organization to get customer feedback. Circulating in front of customers is a vital step for verifying a design’s success and informing future changes, but it adds a distinct demand to the prototyping tool. As Reggie explains:

There are sprint demos and sometimes we have customers maybe not at the sprint demos, but soon after, see the movie or see the actual running code to tell us what they think of the progress and how it’s, the direction its going. (Reggie)

From my observations, it seems that still images, movies, and “walkthroughs” (Reggie; Cathy; Sean) are the most frequent ways that designers use to present designs to customers and other stakeholders. It seems that as the organization has switched to Agile methodology, the opportunity for presenting functional prototypes has decreased. It is rare for customers to directly interact with functional prototypes.

The user experience team considers the non-functional prototypes for user studies as a deficiency, as suggested by Reggie’s hope for even slightly more ‘real’ code. He says:

It would be nice to (gather feedback on functional prototypes because) we’re working to provide more dramatic changes in our UI...and the last time we did that we had mixed results in that the customers who reviewed it thought it was better but that
wasn't the unanimous opinion... and so this time we're working, and development is committed to providing regular prototypes of the interaction such that we can do much more frequent and timely user testing on the actual interactive components. It may not be actual running code, it may be dummy data, but the UI would actually be interactive. (Reggie)

To Reggie, as to Cathy, interactive prototypes are becoming increasingly critical. Reggie states, “In a pure UI design world, having an interactive prototype is becoming essential to make sure the design works.” Or, in Sean’s words, “the best way to do that is, pictures of course are helpful, static pictures are very useful, but even better is an interactive prototype.” Cathy traces the lack of user tests with working prototypes to the switch to Agile, which strives for early user feedback. When asked why she does not do live code tests more often, she says, “because we’re in earlier, so there isn’t working code for things, we end up showing PowerPoint.” By iterating frequently, Agile tries to generate user feedback earlier in the process, when it has a better chance of making a difference, but also when there is less tangible product to actually show. Clearly, the user experience team has identified a need in their collaborative ecology.

The designers in the study identified two solutions to their lack of functional prototyping tools, involve the development team in creating prototypes for test, and appropriate a design tool capable of generating semi-functional prototypes. The first solution, while likely preferable to the members of the study, is rare. In the one case of it (related by Reggie), the development concluded that they would benefit from doing prototypes themselves. Reggie explains:

We're actually trying to contract some folks from china to help with the prototyping and the development team realized that if they do the prototype they're actually going to learn along the way and help guide what’s actually doable, and it’s good news for everyone. (Reggie)
This strategy is consistent with the most idealized concept of Agile and participatory design, but none of the designers seemed to expect it to become common.

The second solution is to appropriate a more sophisticated prototyping tool. By far, the person pushing for a better tool is Sean. His main goal is to communicate designs internally, as he explains:

The collaborative goal here is communicating our design to each other first of all for review and to architects and executives and people like that and then ultimately communicating our designs to the developers with enough specificity that they can immediately grasp what we want them to build and with enough detail that they can understand exactly how every control should work and how to test the functionality to be sure that they've implemented what we've designed for them. (Sean)

He mentioned several options, including a free and open source project initiated by the organization’s software team. However, disregarding the principle of preferring the in-house tool, Sean and his team adopted a commercial tool called Axure, on the grounds that it was a more mature tool with a stronger support community (Sean).

Sean has had some success in mediating his preference for Axure among his own team. Sean works for a related but separate user experience and user interface design team. Some time ago, he and one of his coworkers persuaded their manager to purchase eleven Axure licenses at 500 dollars each (Sean). This is no small expense, and was no doubt facilitated by the large and well funded project that was just starting up in Sean’s group—the same project that allowed him to practice ethnography. He didn’t mention precisely how he convinced the manager, but he did mention a lot of positive reviews from management and others since he began to use it (Sean), and it seems that obtaining management support is a crucial element for mediating costly tools.
Aside from convincing management, Sean also had to convince his teammates to expend time and effort to learn the new tool. New tools like Axure are associated with a learning curve. Even if the tool is available, the designers will only select it if it will pay off despite the time required to learn its use. The high iteration frequency of Agile projects means that the turnaround time for designs is often only a couple of weeks, according to Reggie. There just is not a lot of room in the timeline for learning new tools. As Reggie says:

But there's a learning curve there that many times we just don’t have time for. We have 2 weeks to do a full design, and if it takes a week to learn a new tool that cuts in too much. (Reggie)

Even on Sean’s team, only a couple of workers use it for design—primarily those who lobbied for it in the first place. Most of their coworkers use it to consume designs, make small changes, and provide feedback. Sean admitted that he wasn’t as expert with Axure as he would like. It takes time to learn, but he seemed willing to put in that effort.

Despite the effort required to learn a new tool, eventually the cost of not having sufficiently interactive prototyping will outweigh learning curve and setup costs. Axure seems to be a likely choice. Several of the designers (Ray; Cathy; Reggie) mentioned Sean’s team and their use of Axure when discussing prototyping tools. I observed Sean and his colleague give their first presentation promoting Axure in summer 2011, and by all accounts have they have continued to evangelize. Yet, for whatever reason—funding, learning curve, or something else—members of the other team had not yet appropriated by March of 2012 (Sean), nor had they spent much effort with its free and open source competitors.

It wasn’t until summertime that the other team procured Axure licenses. At the request of the facility’s management, the user experience team agreed to move their usability lab equipment to a smaller space. It was relatively easy to give up the old space—because of
the fewer number of on-site usability activities they rarely needed the physical space, and agreeing to transfer to a cheaper space made room for a massive update to their equipment, including Axure licenses, a license for Morae, which is a commercial tool for conducting and recording usability tests, digital recording equipment, and upgrades more tailored to their current working practices. Obviously, the purchase of these tools represents a categorical selection, and it remains to be seen how smoothly these new tools are incorporated into the collaborative ecology.

This last thread in the user experience collaborative ecology saw several familiar tools emerge, such as MS Word, Team DB, Connector Wiki, shared network storage, emails, and others (Figure 7 next page). Most of these were discussed earlier. What was unique was the variety of design and prototyping tools. MS PowerPoint emerged as an important medium for communicating prototypes. Since team members seem to construct its use as a prototyping tool differently than its use as a communication tool, it appears in the map twice. Photoshop, GIMP, Illustrator, and CorelDraw all showed up as important image editors. Again, repositories and storage solutions proved to be far more political than their unassuming function would suggest, with improvised web pages taking a role as a ‘library’ or ‘repository’ fully controlled by a single graphic artist in order to guarantee its appropriate use. The prototyping tool Axure and its competitors make the most interesting story, as members of the design team seem to desire a prototyping tool capable of generating interactive prototypes, but were not too eager to try a free and open source version, and instead waited until resources became available for an upgrade.
This last thread on prototypes bears a few more insights about selection, practice, and mediation in a collaborative ecology. For one example, Reggie’s use of free and open source software shifts selection-making power from the management class to the individual worker. Also, practice of filtering visual designs into formats that are highly consumable, like PowerPoint, helps to smooth over a lot of local practice, making it possible for individual designers to select image editing tools based on personal preference, which seems to be an institutionally acceptable behavior. We saw improvised sharing and storage tools solely administered by Carol, the graphic artist, giving her almost complete control over her personal portfolio of icons and graphics. Having that control helped her to ensure her work
was used appropriately by others who don’t have her knowledge and history with the activity of visual design. Her tool is both answering a need but also enabling a gatekeeper function. Even though this practice seems unique to her, it can still spread through mediation when she turns control of entire pages over to new administrators. We witnessed an emerging practice that reserves certain storage tools for highly stable designs as Ray doesn’t place early prototypes in Agile Manager because he knows they will be incorrectly perceived as finished. Finally, the section about Axure reinforced some notions of mediation that first came up with regards to Sean’s Team DB. Practical demonstration, timed with a pressing need and ample resources all seem to be required for Axure to take off, but those conditions do not exist for every team in the organization, so mediation is limited.

**Conclusion**

This chapter has explored five threads of activity related to the user experience team I examined. These five threads—the Software Development Lifecycle, User Data, User Requirements, Specifications, and Prototypes—all depend on distinct corners of the broader collaborative ecology. Tracing the five threads has not only identified many of the tools the user experience team needs to coordinate their work, it has also identified efforts of selection, practice, mediation, and how they interact with each other to transform a collection of software applications into a successful ecology The next chapter will discuss some of the implications of these findings.
Chapter 5: Selection, Practice, and Mediation

Chapter four narrated five threads of activity prominent in the collaborative ecology of a user experience team. It emphasized the role of social and collaborative software. The threads that emerged revealed insight into the user experience team’s collaborative ecology, and also illustrated the role of selection, practice, and mediation efforts of the organization. This chapter first discusses some insights about selection, practice, and mediation derived from the five threads from chapter four. Second, it comments on some issues about collaborative ecologies that cut across selection, practice, and mediation. Third, I recognize the limitations of my study. Finally, I speculate on some possible directions for this work.

Three Mechanisms in Five Threads

Examining a user experience team revealed five key threads of activity. Each one was supported by a distinct set of tools and practices, and identifying them produced a sort of map of the collaborative ecology. That map is a valuable resource for understanding the workings of this particular collaborative ecology, but creating it also produced insights into how selection, practice, and mediation assemble and shape that ecology.

Selection

Efforts to select software, both categorical and in given instances, were apparent in the five threads of activity from the user experience collaborative ecology. More importantly, the data demonstrated three further insights about selection that I will describe here. They include a particularly meaningful dynamic between categorical and instance selection, a realization that organizational policy and selection are not locked together, and a tendency for free and open source software to shift selection authority around an organization.
Categorical and instance selection both emerged from the data, and while they played a similar role in assembling and shaping the collaborative ecology, the relationship between the two seems much more complex than I initially anticipated. To review, a categorical selection is one that makes a tool available in a collaborative ecology, while an instance selection is when it is actually used for a task. As anticipated, the data showed that categorical and instance selections were made by different individuals with distinct interests and personal goals. Either type of selection could depend on nuanced and even unconscious reasons. What was surprising was the degree of faith that workers placed in the choices of their colleagues and the relationship between policy and individual choice.

When members of the study tried new tools, their colleagues approached that tool as the presumptive solution for satisfying their own needs. The story about the prototyping tool Axure is a good example. Recall that it was chosen by a team despite the existence of a free and open source package supported by the corporation’s software team. On paper, the free and open source should have been the presumptive first choice. Yet, Sean analyzed it as “less mature” and gave presentations to demonstrate Axure as a preferred choice, eventually persuading two separate teams to purchase licenses. The rest of the team was largely willing to delegate the analysis of which tool to choose and, less obviously, what it’s really good for to one or two trusted colleagues. Largely on account of Sean’s efforts, the tool is available and now paid for, and it apparently satisfies their needs enough that they have little motivation to change behavior, though little data indicates that the actual use of Axure is spreading.

Another way that selection was more complicated than I anticipated was the relationship between policy and individual choices. In a large multinational corporation, I
had expected policy and corporate-level decision making to have a heavy influence on local selections. In some cases it the local workers did follow a receivership model. *Everyone* used the same email system and the same instant messaging system. But wherever it was possible it seemed the corporation supported local choice; the individual workers were not, for example, forced into using the internally developed word processing, spreadsheet, and presentation software. Feedback loops between upper-level decisions and lower-level decisions was an interesting rhetorical dynamics.

One example of the dynamic between local and organizational selections is the role of free and open source software (FOSS). The ongoing success of the open source software paradigm (Mockus & Fielding, 2002, p. 311) and the increasing quality and competitiveness of its offerings quality (Fitzgerald, 2006, p. 587; Ven, Verelst & Mannaert, 2008, p. 54) makes a lot of alternatives available to the collaborative ecology. The reserve of FOSS circumvents typical categorical selections. Provided there are not licensing or security concerns, individual workers are free to choose from a massive pool of tools that are always available, moving the authority for making categorical selections a few steps closer to the local team. Using free and open source software was generally supported by the management class. In other words, it seems that the corporation gains both a productivity and financial benefit from relinquishing this kind of power to local teams.

**Practice**

Like selection, practice was evident in the study in both emergent and conventional forms. The study also implied four insights about practice worth mentioning here. First, the study revealed innovation and improvisation as a way to compensate for a mismatch between
the tools and the activities of an ecology. Second, it identified that a practice can be conventional even for individuals, and can still carry all normative force of more widespread practices. Third, it illustrated a vital dynamic between the desire for uniform ‘best practices’ and the appeal of local flexibility and customizable software. Finally, it revealed that the power of highly stable tools and practices can linger and influence the collaborative ecology long after the organization moves on to a new paradigm of activity.

The first insight about practice that I want to point has to do with innovation and improvisation. At several times in mapping the collaborative ecology I witnessed individuals improvising innovative solutions to particular problems. Examples include Sean’s improvised data analysis tool based on Team DBs, and Carol’s improvised design repository based on a static HTML web space. In both cases, the individual professional took a flexible tool and turned it to an unexpected use, identifying a point where the network of tools does not match the network of practice perfectly. It occurred on an institutional level too. In “bridging” the Agile Manager and Version Control Suite systems, the organization innovated a solution to a problem of practice. I would use the term “improvised,” except for the fact that creating these bridges is the full time work of actual project teams.

There was, of course, another response to mismatches between the tool network and the practice network. Workers lower their standards. This was the case when Cathy expressed a desire for a tool that would smoothly organize and curate old user data. Without being aware of a credible way to solve her problem, she simply did not and could not practice the kind of behaviors she considered correct.

A second insight into practice that arose had to do with conventionality. I had initially assumed that for a practice to become conventional and normative, it would have to begin
spreading. The surprise was that even a single person’s routine practice could become normative. An example is when Cathy and Jim submit new user requirements. While the two of them collaborate on new requirements, and are often in the same room, it is always Jim who personally manipulates the Agile Manager system. Cathy does not participate in that practice, and yet Jim’s convention is expected. In other words, a practice can be individually conventional, expected and normative even by other coworkers who do not participate in the practice. It is akin to saying “he always does it that way.” This individual understanding of convention seems paradoxical and requires further inquiry.

Just like the dynamic between categorical selection and instance selections, there was a certain tension over determining “best practice” and a locally preferred practice. From an open software point of view it is tempting to assume that everyone benefits if they are free to tailor their own local software environment. Draxler and Steven’s (2011) examination of the Eclipse IDE effectively refuted that assumption as a necessary truth, showing that individually tailored software environments can cause a variety of difficulties for collaborative endeavors.

Third, my study revealed both a desire for uniformity and for freedom of individual practice. For example, individual workers lamented their own failure to acquiesce to what they saw as a uniform standard practice when they seemed apologetic for not using wikis or when they asked to have team meetings to establish best practices for using the file storage functions of Agile Manager. On the other hand, the study also revealed a surprising tolerance for local practice, as long as there was a layer of uniform practice insulating the local variation from the rest of the team. It doesn’t matter at all how individual user experience designers created prototypes. As long as everyone maintained the team practice of storing
and sharing designs as a PowerPoint document, no one cared whether your personal practice involved Photoshop or Gimp or Illustrator or anything else. Likewise, if a team wants to use a wiki as a collaborative word processor, they may, as long as they export the finished version to an MS Word file and store it in Agile Manager. Local variation is great, as long as it stays local.

A final insight into practice that emerged was how the normalizing effect of a stable practice can outlast the exigencies that produced it. The main example I have of this is the usability facilities maintained by the user experience team. Prior to the shift to Agile methodology usability efforts centered on a testing phase that occurred after development. Consequently, the team maintained a large usability lab with one way mirrors and recording equipment and lots of physical hardware. Under Agile, the team’s usability efforts have shifted to the design phase of a project, and have become more iterative. The team continues to see the kind of data produced by such tests as desirable, even as they do more work remotely and with mock designs rather than functional prototype testing.

I need to be clear here; the older model is not obsolete. The practices that stabilized under the previous model are influential and remain influential for good reasons. What I want to observe is that those standards and practices remain influential even under the newly forming model, despite the fact that the team has not yet fully integrated the best of both models. They still seek a balance, and that balance will incorporate a lot of the older model.

**Mediation**

Like selection and practice, mediation efforts were also apparent in the user experience team’s collaborative ecology. While mediation is a fair degree more complex than
selection or practice as they are currently envisioned, the data still provided a few important insights in its regard. First, mediation depends on timing and opportunity. Second, mediation is associated with influence, especially as it accumulated in networks surrounding innovators, trendsetters, and leaders. Third, the study suggests that the dichotomy between formal and informal mediation is more of a continuum, suggesting that formality may not be the most useful way to dissect efforts of mediation. Finally, the data suggests that mediation is not exclusively a human endeavor, but that tools can advocate and mediate themselves.

The first insight into mediation that arose from the study was the importance of timing and opportunity. For changes to mediate through the collaborative ecology certain requirements had to be met. The best example of this is, again, the Axure case. Sean had convinced both teams that Axure was the best solution to their prototyping issue, and yet for almost a year the use of the tool did not spread to the other team. The reason partly had to do with funding and partly to do with the time it would take for each individual to follow their own learning curve. In Sean’s team, the opportune moment was a large, original project with a lot of extra funds and time. In the other team, the opportune moment was a year later when they accepted a reduction of physical resources in exchange for new software. In both cases, an event in the collaborative ecology caused an opportunity for a tool use to mediate and become available in another part of the organization. It also seems that the timing of mediation depends on successfully meeting selection and practice requirements. If a new tool and way of working does not seem useful, or does not turn out to actually be useful, then its distribution in the collaborative ecology cannot be stable and must shrink.

A second insight into mediation has to do with personal influence. Mediation depends on an accumulation of power. If we think of Reggie and Sean, both seemed to have more
influence over their collaborative ecology than other members of the study. They also had more willingness to try new ways of working, they had more expertise with a handful of tools that they leveraged to influence their collaborative network, and, they were both generally considered authoritative and trustworthy individuals. Even though they don’t always advocate the same tools or the same way of working, they both accumulate credibility in their collaborative ecology.

A third insight about mediation that arose in the study is that formal and informal mediation operate as a continuum rather than a binary. Chapter one anticipated that mediation would be both a formal and an informal activity. Defining the difference qualitatively is easy. A training session or a document obviously has the character of an intentional, formal, planned effort of mediation. Walking across the hall to ask for help on a software task obviously has the character of an improvised, last minute, informal effort. But there is no obvious distinction between where one kind of mediation ends and another begins, and there is no obvious distinction about what sorts of stakeholders employ what kind of mediation effort and when. More commonly, mediation depended on a collection of variously formal and informal efforts.

A fourth insight into mediation is that it is not something that humans do exclusively, confirming the usefulness of the ecology metaphor and adding support to actor-network theory’s principle of human and nonhuman symmetry. It is relatively simple to tie individual persons to selections and practices. Even in a sociocultural understanding of those two concepts, it seems that a person naturally injects intention into the dynamic. It’s not the case with mediation. As I wrote before, some tools evangelize for themselves, without the need
for a human person to actively pursue mediation. People and tools are co-constitutive, and both play a role in assembling and shaping the collaborative ecology.

Because of this, mediation is distinct from selection and practice. I can suggest two avenues for explaining this curiosity. One way is to draw a point of contact with the affordance concept of usability, where the outward features of a tool indicate to a person the ways that it might be used. Even though the features of a tool are the result of a design activity, affordances places the tool in the subject of the mediation activity. Another way is to flatten the distinction between person and tool, following the actor-network approach of treating humans and nonhumans symmetrically as actants.

**Beyond Selection, Practice, and Mediation**

The concepts of selection, practice, and mediation seem to be both evident from and useful for understanding the data. This section raises some issues that emerge from the data but go beyond the concepts of selection, practice, and mediation as currently envisioned.

The first issue in the study that emerges beyond the selection, practice, and mediation division is ideology—in particular the ideology related to Waterfall and Agile software management practices. The term “ideology” really is unusual in a study that is methodologically grounded in actor-network theory. Its referent should instead be considered a remarkably stable and powerful network that extends itself over a huge group of humans and nonhumans, constraining them into a particular role.

Labeling these particular networks as ideological still seems apt, since they exhibit most of the qualities that the term connotes. An ideology is “a system of political or social ideas, framed and propounded for an ulterior purpose,” where “the ideas are so related that
they have in them, either explicitly or implicitly, inducements to some social and political choices rather than others,” (Burke, 1955 p. 88). In short, Agile software principles, but also lingering principles from Waterfall, play a tremendous role in shaping the way that study members perceive the collaborative ecology that they inhabit by acting as an ideology.

I see two possible avenues to resolve this concern. One way would be to develop selection, practice, and mediation as concepts for dealing with intellectual tools. While I have taken care in this project not to exclude such an application, I have not explicitly pushed the analysis in that direction. Another way would be to direct an analysis at a more global level. As I have discussed selection, practice, and mediation so far they are very local concepts, attached to particular people using particular purposes in particular situations. In the future, they could explain more global concepts like aggregations of these small local connections to make up a network of ideology.

A second issue related to the role of ideology is the notion that selection, practice, and mediation produce a large network in aggregate, but that is made up of smaller networks. Consider again the tolerance for local practices that I mentioned earlier. So long as a local practice stays local by not interrupting the work of the organization on a larger scale, it is tolerated or even encouraged. But when we back away from looking at a small region of a collaborative ecology and start looking at large regions a new story emerges.

Take, for example, the large ecologies around the competing software development life cycle managers, Agile Manager and Version Control Suite. While one seems to be privileged from the point of view of my participants, both are remarkably large, stable networks, buoyed by ideology, by best practices, and by history, and all sorts of other concerns. These networks have accumulated a lot of power and are consequently quite stable.
They are each built of smaller networks: an ecology for user experience, one for developers, one for testers, or however else you might decide to draw boundaries. And, between the two there is a boundary area where workers and tools and practices arise that try to live in both worlds at once. Those boundary objects have a “common identity across different sites,” (Star and Griesemer, 1989, p 393.) Drawing on this idea, Wilson and Herndl write, “This common identity offers a way for the different participants to occupy a shared space that accommodates their differences and their common purpose or motive,” (2007, p. 138), linking Star and Griesmer’s analysis of an object’s plasticity and robustness to Burke’s (1969) rhetorical tension between division and identification. In my study, this view of boundary objects seems to apply to individuals and practices as well as to artifacts, since workers like Jim are “being torn apart” by having to maintain many different competing practices, and since software teams have to develop “bridges” between the two massive networks of practices.

A third issue that seems to slice across selection, practice, and mediation is the idea of stability. In tracing the collaborative ecology of the user experience team it was a simple thing to identify stable and unstable tools and practices, competition and discrepancies of tools and practices, and actions of appropriation and accommodation. This clarity didn’t arise from thinking about either selection, or practice, or mediation alone but about all three in general.

One way to think about the stability of a tool or practice is to imagine how much trouble it would cause to try to eliminate it. At a few points in the case narrative, I mentioned tools that were in a phase out period, where a legacy solution was supported but discouraged. Phase out periods are a testament to just how stable a set of tools and a way of working can
become. Less stable tools and practices (like selecting one of the many available web conference system) are swapped in and out of the collaborative ecology casually.

I have two comments about the stability of technology uses. First, stability, like everything else, is a locally contingent phenomenon. What is stable in one part of the ecology may not be stable elsewhere. Second, understanding selection, practice, and mediation may be able to contribute to an understanding of stability generally. Selection can help explain how well tools fit and therefore enter the ecology, practice can help explain how their use stabilizes, and mediation can help explain how their use spreads. All three are required to a degree, each one contributes to the overall stability of a tool and practice, and the more evidence we see of each presumably increases our understanding of how stable the tool use is.

Like stability, another phenomenon that seems to cut across selection, practice, and mediation is redundancy. I was surprised to see how many redundant tools the collaborative ecology maintained, especially for file sharing. From the beginning, the sheer number of places to store files seemed like a digital labyrinth with few hints for navigate. But as the study produced data it became apparent that redundancies were actually a useful feature. Having half a dozen competing tools for ostensibly the same task ended up suggesting finer nuances in what the study participants were up to. Having the freedom to choose one storage location for its security and another for its accessibility, and another for its simplicity, and another because it communicates authority certainly complicates the overall collaborative ecology, but that complication merely reflects the complicated nature of what the workers are aiming to do. The tools were not redundant at all.
The variety of storage tools and the benefit of ostensibly redundant tools in general illustrate an important feature of the overall collaborative ecology. The classes of software that constitute the collaborative ecology vary greatly along several dimensions. Some tools were very open, allowing for a lot of freedom to innovate and improvise. Other tools were closed, rigidly enforcing patterns of practice. Some tools were very specialized and deeply linked to a particular niche of the organization’s activity. Some had an almost universal application for members throughout the organization. And, some tools were formal and had a character of credibility of their own, while others were informal and relaxed.

By maintaining a collection of software tools with such wide-ranging attributes, the workers were able to distribute their efforts. Tools are not just used because they fit our purpose perfectly. Sometimes they are used because they can be adapted to new purposes, or because they make your work seem more official, or because you are more comfortable with it, or because you know another team will not see your work prematurely. An example again comes from the Axure case. Axure has the capability to render a prototype as a wireframe with squiggly lines that look hand drawn. The feature does not have a functional purpose, yet it was still a selling point because it could indicate that the design is a work in progress. Having a mix of tools that differ in their openness, specificity, and formality enables the workers to strike a balance between enforcing patterns and working flexibly.

**Idiosyncrasies of the Study**

Owing to the design of the research, this study has some idiosyncrasies. Some obvious and expected ones, like the inability to take formal field notes, and the need to pseudonym participants and some software applications, were discussed in the methodology.
chapter. I want to point out two further issues. First, the broad scope of the research prevented me from closely examining particular tools, which was expected but turned out to be more prominent than anticipated. Second, the organization distributed information technology support duties to members of particular teams, which is not a universal approach and has consequences for the study.

In this project, I made an intentional effort not to examine a single tool, but to examine the mechanisms that assembled and shaped many tools, practices, and language into a collaborative ecology. I stand by that decision as a novel way of thinking about the shared use of collaborative and social software, because it produced a lot of interesting findings that could not have arisen otherwise. But it did prevent me from examining particular tools in detail.

The reason it is important to examine tools, or practices, or language at a fine detail is because individual people often make finely nuanced local decisions. Delving in to those nuances and examining them would be a fruitful way to understand their design and use, and why one tool can stabilize in place of another. But the broad scope of my work prevented some of that kind of analysis. The problem was exacerbated by the fact that I could not reveal proprietary names, analyze interfaces, do usability tests and design critiques, and otherwise examine what was going on at a fine grain. These kinds of data gathering were at odds with my project’s design.

Second, the organization I studied was potentially atypical in that it assigned information technology support duties to members of local teams rather than having prominently information technology department. This has important implications because I.T. traditionally plays a role in administering the balance between locally tailored tools and
practices and organization-wide uniformity. In this study, the absence of formal IT seems to lead to a lot of liberty and self-determination. A system-focused IT presence might lead to a more rigid activity, while an ecology-focused IT presence might play a guidance and facilitation role. Examining how IT might be conducted is an interesting avenue for further research.

**Ecology Design**

So far in this last chapter I have pointed out some insights about how collaborative ecologies are assembled and shaped. Many of those insights are linked to selection, practice, and mediation. Others insights cut across them or are separate from them entirely. Speculating on these insights and on the project as a whole I want to recognize an oncoming paradigm in design.

In “The ecology of writing,” Marilyn Cooper interpreted the idea of ecology as an alternative to the solitary writer proposed by cognitive process models. She writes, “The ideal image the ecological model projects is of an infinitely extended group of people who interact through writing, who are connected by the various systems that constitute the activity of writing,” (1986, p. 372). Ideas are part of a “landscape that is always being modified by ongoing human discourse,” (p. 372). Articulating writing as an ecology allowed for understanding communication across a broader set of dimensions that the individual writer’s cognitive process. Of course, flattening a dynamic system by focusing on a particular dimension, as the cognitive process model does for writing, is a useful way to simplify complex challenges.
Just as writing is a dynamic, social interaction of infinitely extended groups, collections of contemporary social and collaborative software are dynamic sociocultural systems that involve people, activities, tools, and ideas. Ongoing efforts to simplify this chaos include approaches like content strategy. A unified content strategy, as defined by Rockley, Kostur, and Manning, is “a repeatable method of identifying all content requirements up front, creating consistently structured content for reuse, managing that content in a definitive source, and assembling content on demand to meet your customer’s needs,” (2003, p. 12). While content strategy offers a useful way to simplify and manage the challenging dynamics modern information systems, the exploration of collaboration as an ecology in this project suggests an alternative approach to managing and designing social and collaborative software systems.

This project has not designed anything itself, but design is involved nonetheless. In a sense, the entire project has circled around the idea of design as understood in light of the sociocultural turn in several fields. As an example, consider these three quotes:

Usability cannot be an attribute of a thing, but rather must be a relation between a user and an artifact as embodied within a set of practices. (Zemel et al., 2008, p. 63)

I examine the crucial subversive interactions in which workers routinely engage as they use information systems to accomplish their activities....My goal is to better understand…why workers so often alter the designed artifacts (particularly textual artifacts) they are presented with, and how designers might approach design tasks as true partnerships that result in designs flexible enough to be adopted. (Spinuzzi, 2003, p. 4)

Users are designers (Norman, 2004), who are actively redesigning, or-more accurately-localizing, an available technology to fit into their local contexts. ... they know what works in their own contexts, and they know how to make use of a technology in their life spheres if they are able to find a good fit. (Sun, 2006, pp. 458-9)
These three quotes suggest a new approach to technology where the responsibilities for design don’t end when a product is released, but instead transfer from the developer to the user. And while it’s true that some users can crack open the code and manipulate a piece of software, the role that end users play can be seen much more readily as influencing the ecology than as influencing the code. Rethinking design for ecologies seems to be the next logical step.

In designing for ecology, users are clients rather than customers. Software applications are contracted and released like consultants. The humblest stopwatch widget and the most sophisticated software IDEs and SDLC managers are building blocks in an ecology, to be inserted or ejected as needed, and to be combined with improvised practices and systems of work. Of course they matter; they are part of what constitutes and shapes the ecology. But they are not permanent and their role is not fixed. And we have made precious little progress in learning how to facilitate that kind of design, either before or after software release.

This project has been fortunate to see both sides of the design divide: design in the code and design in the ecology. On one side, it observed a user experience team trying to design a usable software product, and employing some of the best industry practices to do so. That same team is also a group of users, assembling and shaping a collaborative ecology of their own through making selections, developing practices, and mediating knowledge. The study’s main contribution has been a map of that collaborative ecology that I believe is already a useful tool for navigating and facilitating that ecology.

At the same time, I think the strategy has further promise. In particular, I believe the approach’s ability to observe and perhaps crudely measure the stability of particular tools and
practices could prove very useful in ecology design. Following that belief and refining the approach will require a lot of future work, but it is the kind of refinement that the current climate of social and collaborative software demands.
Appendix A: Letter of Informed Consent

INFORMED CONSENT DOCUMENT

Title of Study: Negotiating collaborative technology in professional work

Investigators: David Niedergeses

This is a research study. Please take your time in deciding if you would like to participate. Please feel free to ask questions at any time.

INTRODUCTION

The purpose of this study is to learn about how professionals negotiate the selection of collaborative software packages and how they develop conventions for using those packages. You are being invited to participate in this study because you are member of a group of professionals who are using collaborative software in novel ways. You should not participate if you are under the age of 18.

DESCRIPTION OF PROCEDURES

If you agree to participate, you will be asked to participate in an interview about your organization’s use of collaborative software. Interviews will take between 30 and 90 minutes at a time and place of your convenience which will be recorded in digital audio format. In addition, the researcher will take written observations of your day to day behavior in meetings and interactions with other collaborators to be discussed during follow up interviews.

All interview questions and observations will be strictly related to your professional opinions, attitudes, and behaviors and how you coordinate work with coworkers using collaborative technologies.

RISKS

While participating in this study you may experience the following risks:

There are no foreseeable risks involved with participating, apart from the minor chance that you may feel uncomfortable being interviewed and observed. You may opt out of all or part of the study at any time for any reason.

BENEFITS

If you decide to participate in this study there will be no direct benefit to you. It is hoped that the information gained in this study will benefit society by advancing knowledge about the behavior of collaborative groups using social software, about how collaborative groups are managed in open-ended software environments, and about how to improve the technologies they use.

COSTS AND COMPENSATION

You will not have any costs from participating in this study aside from the time required to participate. You will not be compensated for participating in this study.
PARTICIPANT RIGHTS
Your participation in this study is completely voluntary and you may refuse to participate or leave the study at any time. If you decide to not participate in the study or leave the study early, it will not result in any penalty or loss of benefits to which you are otherwise entitled. You can skip any questions that you do not wish to answer.

CONFIDENTIALITY
Records identifying participants will be kept confidential to the extent permitted by applicable laws and regulations and will not be made publicly available. However, federal government regulatory agencies, auditing departments of Iowa State University, and the Institutional Review Board (a committee that reviews and approves human subject research studies) may inspect and/or copy your records for quality assurance and data analysis. These records may contain private information.

To ensure confidentiality to the extent permitted by law, the following measures will be taken: All audio recordings, transcripts of interviews, field notes, and other documents will be stored on a password protect computer until the project concludes in August 2012. Thereafter, all data will be digitally encrypted and archived. All references to individual and organizations will use pseudonyms in any publication. If the results are published, your identity will remain confidential.

QUESTIONS OR PROBLEMS
You are encouraged to ask questions at any time during this study.

- For further information about the study contact the principal investigator, David Niedergeses, 515-709-1957, dniedergo@iastate.edu or the project supervisor, Gregory Wilson, 515-294-2180, gwilson@iastate.edu.
- If you have any questions about the rights of research subjects or research-related injury, please contact the IRB Administrator, (515) 294-4566, IRB@iastate.edu, or Director, (515) 294-3115, Office for Responsible Research, Iowa State University, Ames, Iowa 50011.

******************************************************************************

PARTICIPANT SIGNATURE
Your signature indicates that you voluntarily agree to participate in this study, that the study has been explained to you, that you have been given the time to read the document, and that your questions have been satisfactorily answered. You will receive a copy of the written informed consent prior to your participation in the study.

Participant’s Name (printed) ________________________________

(Participant’s Signature) ____________________________ (Date)
Appendix B: Interview Protocol

**Introduction**
1. Please describe what you do for work, especially the parts where you interact with others?

**Tools**
2. What kinds of technology do you use to collaborate with your coworkers? Describe some.
3. For a particular example, why do you use it? How does it help you? Are there other tools that might work? Why not those?
4. Can you describe the process of how this tool was selected for these tasks? Who was responsible? What was considered?

**Your use**
5. Can you describe HOW you use the tool? Is this the best way? How else could you use it? What are the benefits and drawbacks of this practice?
6. How did you and your team arrive at this use? If you could, what would you change about how this tool is used?

**Others’ uses**
7. Do you and other members of your organization always use the tool this same way, or is there variation? Can you explain this variation, or lack of variation? What are the consequences of that variation?

**Learning**
8. Tell me about what it was like learning to use the tool in this way? What helped you?
9. How would a new user learn the appropriate way to use this tool? If you were responsible for teaching them, what would you do?

**Relation to other tools**
10. How does this tool relate to the other tools you use? What role does it play in your overall collaboration? How did you arrive at that role?

**Conclusion**
11. Please expand on your answer to question #X.
12. Of what we’ve discussed, which topic is most interesting to you? Do you have more to say on anything?
13. Can you think of anyone who would have an interesting view on this kind of material? Someone I should talk to?
References


Biographical Sketch

David Niedergeses was born in Omaha, Nebraska on May 5, 1982. He lived there with his family until enrolling at Iowa State University in 2000. After graduating with a bachelor of science in electrical engineering and a bachelor of arts in English in 2004, he continued in Iowa State’s rhetoric, composition, and professional communication program, receiving a master of arts degree in 2007. Then, he moved into the university’s doctoral program with co-majors in rhetoric and professional communication and human computer interaction, which he completed in the summer of 2012. During his time at Iowa State University, David participated in numerous internship and co-op positions with companies including MTEC Photoacoustics, Carleton Life Support Systems, EFCO Corporation, and most recently with IBM, where he will be taking a position as a user experience designer in the fall of 2012.