A. Cover Page

NSF ITR Research Proposal

ITR/SOC: Understanding Open Software Communities, Processes and Practices: A Socio-Technical Perspective

NSF Award - #0083075

PI: Walt Scacchi
Co-PI: Mark Ackerman
Institute for Software Research
Information and Computer Science Dept.
University of California, Irvine
Irvine, Ca 92697-3425 USA
949-824-7355, 949-574-0481
949-824-4056 (Fax)
Wscacchi@ics.uci.edu
Ackerman@ics.uci.edu

Relevant NSF ITR Programs:
I. Social and Economic Implications of Information Technology
II. Software Processes

B. Project Summary

Our interest is to develop empirically grounded models and theory of the social processes, technical system configurations, organizational contexts, and their interrelationships that give rise to open software [cf. KS82]. "Open software", or more narrowly, *open source* software, represents a new approach for communities of like-minded participants to develop software systems and related utilities that are intended to be shared freely, rather than offered as commercial products. While there is a growing popular literature attesting to the potential or apparent success of open software, there is little in the way of careful systematic empirical study that characterizes or informs: (a) how such communities produce software; (b) how do they coordinate software development work across different settings; (c) what social processes, work practices or organizational contexts constitutes how open software is produced and sustained, and the like. However, researchers in a number of distinct scientific/engineering disciplines and commercial enterprises appear motivated to adopt open software practices as a basis for reducing the complexity and improving the reliability of software programs relevant to their domains. Thus, to the extent that science research communities or commercial enterprises seek to primarily follow popular prescriptions and testimonials regarding the efficacy of open software, we cannot be assured that their activities will be most effective or efficient investment of their time, skill and related resources.

A comparative empirical study of open software work structures, processes and practices from socio-technical perspective is most appropriate. Such a study must concurrently examine the social arrangements, work processes and practices from which open software communities and artifacts arise, as well as the technical information infrastructure through which these communities and artifacts are articulated and shared. Studying either the social or the technical arrangements in isolation would discount the significance of how the continual emergence of both is a jointly situated socio-technical ecology. Such discounting could easily facilitate misunderstanding and unwise investments.

We will investigate, compare and contrast four communities engaged in the production, use and evolution of open software. Two communities are engaged in scientific research: in high-energy astrophysics (with the Chandra Observatory [C00]), and in software architecture (within the larger Software Engineering community [SoAR00]). The other two communities are engaged in developing and sustaining the Apache Web server [F99], and in enhancing the computer action game, *Unreal* (a global community spanning over 100 Web sites [U00]). These last two communities are primarily engaged in software development, not research, but have relationships with commercial enterprises.

We have developed a research approach that entails the investigation and understanding of representative socio-technical phenomena and conditions using multiple, comparative analysis techniques. We will employ field study methods to examine each community, and case study methods to examine and compare selected phenomena or conditions within and across communities. These techniques are employed to increase the generalizability of the findings and results we produce. Thus, our research purpose is primarily exploratory and observational, rather than aimed at experimental testing of existing theory or orthodoxy.

C. Project Description

The research project we propose can be described in terms of our research objectives, related research, our research approach and anticipated outcomes. Each of these is addressed in turn following an introductory overview.

C.1 Background Overview

Our interest is to develop empirically grounded understandings of the social conditions, work processes and practices, technical system configurations and organizational contexts that give rise to open software [KS82]. "Open software", as epitomized by recent public attention to so-called *open source* software, is said to represent a new or revolutionary approach for communities to develop software systems that are intended to be shared freely, rather than offered as commercial products [DOS99].

Early examples include the Berkeley Software Distributions ("BSD") of the Unix operating system and related utilities [McK99], while more contemporary examples can be associated with organizations like the Free Software Foundation [S99], the Apache Project [F99], and many informal and formal enterprises surrounding the Linux operating system [T99]. Alternatively, organizations like the Internet Engineering Task Force (IETF), the World-Wide Web Consortium (W3C), the Open Science Project [OS00], and the National Institute for Standards and Technology represent loosely-coupled (IETF, OSP) or institutionalized (W3C, NIST) enterprises with interests in open software. These enterprises facilitate the development and sharing of software specifications, privacy guidelines, data exchange protocols or notations, etc. This diversity of organizations and interests are why we choose to draw attention to "open software" rather than just "open source code" programs [cf. DOS99].

While there is a growing popular literature attesting to the success of open software, there is little systematic empirical study that characterizes or informs questions such as: (a) how do such communities self organize or coordinate to produce software? (b) what organizational forms are commonly used to structure software development practices? (c) what social conditions, work processes and practices, and organizational contexts constitute an open software community? (d) how do such communities deal with problems arising in their software development practices? (e) what kinds of changes do participants in a community seek to resolve such problems? At present, we lack the scientific knowledge needed to answers to questions like these. Nonetheless, scientific researchers and other participants in disciplines like Astrophysics, Genomics, Medicine and Software Engineering appear motivated to adopt open software practices as a basis for reducing the complexity and improving the reliability of software programs relevant to their research domains. Thus, to the extent that science research communities seek to primarily follow popular prescriptions and testimonials regarding the efficacy of open software, we cannot be assured that their activities will be the most effective investment of their time, skill and related resources, nor guarantee quality of the software they build.

We believe that a comparative empirical study of open software communities from socio-technical perspective is required and appropriate. We believe that such a study must concurrently examine both (a) the social arrangements, work processes and contexts from which open software communities and artifacts arise, as well as (b) the technical information infrastructure through which these communities and artifacts are articulated and shared. Studying either just the social settings or the technical artifacts and information infrastructure in isolation, we believe, would likely discount the significance of how the continual emergence of both is a jointly situated socio-technical ecology. Why is this so?

Suppose we were to focus attention exclusively on the technical artifacts (e.g., source code programs, documentation, and related Web site contents) as our basis for empirical inquiry. This is something akin to the perspective appearing in popular treatments of open software. Accordingly, this might lead to examining how open source software systems are technically superior to proprietary, closed source, or "Cathedral"-like software systems because open source software must pass the testing and engineering scrutiny that results from public reviews and acceptance in a communal market or "Bazaar" [cf. DOS99]. However, what such a perspective discounts or misses are the underlying social conditions, work practices, organizational processes and institutional contexts that constrain/facilitate which open software artifacts, information sharing practices and community information infrastructures will flourish, succeed, fail, or transition to commercial endeavor. Conversely, to focus exclusively on the "social implications" of open software movements and communities may only modestly inform how scientific communities new to open software might best develop their domain-specific software systems, utilities, documents, related information artifacts and community information infrastructures in an effective (reliable) and efficient (complexity reducing) manner. Subsequently, a line of inquiry that concurrently investigates, characterizes and models the configuration of technical artifacts, social conditions and work processes, as well as the interleaved historical and institutional contexts in which they are situated, is needed. Thus, we believe an empirically grounded, socio-technical study of open software should be the most efficacious path to pursue.

As such, we propose to conduct a comparative study of open software communities, processes and practices from a contemporary socio-technical perspective. Such a perspective follows an emerging tradition of social, historical and ethnographic studies of science/technical work juxtaposed against the technical artifacts or systems that result from such work [BHP89,BST97, K-C97,KS82,L88,St93,SR96,Su87,W98]. However, though economic notions like the theory of public goods [Sa54], collective action and the "free rider" problem [O71] may inform our analysis, we are not primarily addressing economic issues or consequences associated with open software. Nonetheless, such study the economic implications of open software is needed.

We plan to collect and analyze data using from two open software communities outside of traditional scientific communities, as well as two scientific communities that have open software activities underway. The two communities not directly involved in scientific research are the Apache Project [F99] responsible world's most widely used Web server, and the Unreal computer game world [U00]. In contrast, the two scientific communities are the high-energy astrophysics community using the Chandra X-ray Observatory satellite [C00], and the loosely defined Software Architecture research community [cf. SoAR00], supported by Darpa and other research agencies. Our choice of these communities for study is explained later (cf. Section C.4.1). Data will be collected using grounded theory techniques [GS67]. Our use of these techniques includes structured interviews, participant observation, examination of artifacts/products, Web-based survey questionnaires, and the like. We anticipate conducting interviews a minimum of 10-20 participants from each community each year, supplemented with Web-based questionnaire data from 10s-100s of participants dispersed across each community.

Data such as these will then be described, codified and represented in narrative, semi-structured, and computational models in ways suitable for distribution and sharing on the Web. Furthermore, we expect that data collection and analysis will be ongoing and iterative, with three iterations (corresponding to one iteration per year for three years) being necessary and appropriate for longitudinal study.

C.2 Objectives

We have two kinds of objectives to realize in the proposed effort, *research* objectives and *programmatic* objectives. Each is described in turn.

C.2.1 Research Objectives

We have five research objectives. First, we seek to understand how open software communities operate, what work processes and practices they use to produce and evolve software, and how they build software that is reliable in their view. Second, we intend to develop such understanding by conducting multiple field studies that give rise to case studies that can be analyzed and compared within and across open software communities. This theme of comparative analysis of field study and case study data/artifacts is also carried through our third objective: to develop computational, semi-structured, and narrative models that support and embody the comparative analyses we seek. Our approach to comparative analysis and model development is designed to be iterative, incremental and longitudinal over a three year study. Comparative analysis is a hallmark of qualitative field study [GS67] and case study [Y89] research. It seeks to increase the generalizability of qualitative research results. It therefore serves as the foundation for our fourth objective. This is to develop the foundation for an empirically-grounded theory that characterizes, explains and begins to predict how open software communities operate through alternative work structures, processes and practices to produce reliable software systems for community participants. Our fifth/last research objective is to produce and disseminate the anticipated results of our research in a number of outlets and venues (cf. Section C.5).

Overall, the central thrust of our research objectives is to be exploratory and empirically grounded. We do not seek to substantiate or refute existing normative models, theories or frameworks for how open software communities should work, nor whether their processes, practices and technical systems are consistent, complete or otherwise virtuous in some engineering sense. Instead we are interested in eliciting, capturing, coding, modeling and analyzing the collective actions of open software community members for how they produce, sustain and evolve complex software systems, artifacts and their embedding community of practice. Accordingly, such an interest is best addressed by a research team that is skilled in social/organizational analyses of computing work environments, and knowledgeable about the domains of software and software engineering. In being so prepared, we are therefore positioned to jointly contribute to the growing base of science that is empirically informed by studies of social and organizational processes, work practices, computer supported work environments, and software engineering.

C.2.2 Programmatic Objectives

We recognize that NSF through its ITR initiative to which this proposal is directed has identified a number of programmatic objectives. Our research effort addresses topics within the scope of

(a) Social and Economic Implications of Information Technology and (b) Software. Accordingly, we can briefly address the kinds of programmatic objectives we seek in each area.

For *Social and Economic Implications of IT*, we identify four objectives that align with those in the ITR Program Solicitation. First, we seek to understand the particular socio-technical values that are embedded in the development processes and products of open software communities. We seek to model and comparatively analyze the development and evolution of open software and the communities that produce them. Third, we seek to understand how open software technologies (or techniques) evolve to better meet the socio-technical requirements of disparate communities. Fourth, we seek to understand how diverse communities of practice and sustain the distributed and local use of information infrastructures and networks over a period of years through an iterative, longitudinal study. While these objectives do not directly address the economic implications of open software, we believe our programmatic objectives do nonetheless represent an important step in understanding the socio-technical implications of open software.

For a *Software* perspective, we identify two objectives that align with those in the ITR Program Solicitation. First, we seek to increase the scientific basis for software engineering through empirically grounded, comparative field studies and comparative case studies of open software development work structures, processes and practices. Second, we seek to characterize, explain and model how open software communities build reliable software systems through the socio-technical work structures and processes they enact within their community of practice.

Therefore, with these research and programmatic objectives in mind, we turn to describe related research that informs our research approach.

C.3 Related Research

We find five areas of related research bear on the problem area and research approach we propose. In each area, we *highlight* analytical themes we can explore through field study research methods described in Section C.4.

C.3.1 Coordinating software production work

What is the best way for *how to coordinate software production in an open software community?* From a social perspective, scholars like Kling inform us that *cooperation and social control* are simultaneously at play in collaborative software development efforts [K94]. To help mitigate against a tendency for hierarchical control, *shared leadership* has been proposed for use in open software projects [F99]. However, whether this replaces hierarchy with oligarchy is unclear. Similarly, both *formal and informal communication modalities* (e.g., via software review meetings vs. email, respectively) are needed to coordinate software development [KS95]. Subsequently, one place where some form of cooperation, shared control, as well as formal and informal communication are likely to be at play in an open software development project is in *software configuration management activities* [cf. G96,NS87,NS97,NS99]. SCM addresses the provision of *software utilities and workplace procedures that seek to coordinate the asynchronous modification of shared software code and artifacts* [So95]. Subsequently, SCM practices are a likely arena where socio-technical dynamics will come into readily observable places and workspaces [G96,NS99]. In contrast, technology centric approaches stress the need for explicit *representations of links* (or relations) *between software development products,* *organizational processes, information workspaces, and people* to help coordinate distributed software development projects [FW+98,MS90,MSM95,MS96,NS99]. Finally, other efforts seeking to span the socio-technical spectrum suggest providing explicit representations and support systems to help *how participants analyze, diagnose, coordinate, and share experiences in the repair of work processes that have failed or broken* during their enactment [MS93,S98,S99,NS97, SiMG99], as well as *how they collective act to innovate (or redesign) their work flow, content or experience* in response to *changing conditions, events or emerging technical opportunities* [cf. BS89,JS95,KS82,SN97,S00,S00b,VS99].

C.3.2 Organizational forms and work structures for developing open software

What is the most appropriate way for *how to structure the organization of software development work* within an open software community? Starting with Conway's conjecture more than 30 years ago, *does the structure of a software system (its architecture) reflect the organizational form which produce it* [C68]? If so, why? Following [KS82], *if open software systems are a form of social organization*, developing such systems is tantamount to embodying a reflection or image of the organization that produced them within the systems themselves. Is this any different from saying that the "engineering project" is the most common organizational form for coordinating software development work [S84,BS96]? Yes, in that *multiple alternative organizational forms exist for large projects* [GHP99], while *multiple shifting patterns of work organization typify small group effort* [BS89] or "extreme" [B00] development efforts. Similarly, at the organizational level, can we determine whether open software projects are most frequently organized as a *joint venture* [ID98], multi-organization *network alliance* [C99,KoL99,MF+98], or *virtual enterprise* [FW+98,K99,NS99, SB98]? To address such concerns, we need to identify and characterize *the organizational forms or work structures that participants enact when developing open software systems*. Similarly, we need to address *how/if such forms are embodied in the software's architecture*, as well as *how/if they are implicitly embedded in open software system-user behavior*. Conversely, we need to examine *how/if the architecture of open software systems facilitates/constrains the organizational forms that emerge to produce and sustain them*.

C.3.3 Information sharing and organizational memory in a community of practice

How does a participant in an open software community of practice determine who knows what, where/how to seek shared information, or whom to seek for help, guidance or support [AH00]? *Communities of practice* (or "social worlds" [BST97,St93]) are groups of *people who share similar goals, interests, beliefs and value systems* [W98]. In pursuit of these goals and interests, they employ common practices, work with the same tools, and learn how to express themselves in a common language [BD91,LW91,GIK95]. As such,participants need to learn to recognize and *understand how community knowledge of work practices and shared artifacts is distributed* (among few or many), *multivalent* (of mixed provenance), *situated and boundary spanning yet heterogeneous*, rather than centralized, coherent and functionally homogeneous [AH00,BST97, LW91,SR96, Su87]. Further they must understand *how such shared knowledge is transformed, enacted and dispersed through recurring or emerging open software work processes* [AH00,BD91,CoB99, GIK95,SM97] *artifacts and technical systems* [MCM95,Sh91], rather than simply cast in documents or procedural guidelines stored in shared online repositories.

C.3.4 Human-centered studies of computing-based work environments

For almost two decades, human-centered studies of computing-based work environments have shown how *computing system design is also a design for work practice and social organization* [e.g., BS89,KS82,Su87,Su95]. In contrast, the traditional approach in engineering design, particularly within a software engineering approach [So95], is to treat the work practices of the system developers, users and maintainers as ancillary, functional or rationally purposeful. Thus the *organizational, social and technological contexts that surround and embed software work activities* are generally unaccounted for and missing [cf. S84,JS95]. Subsequently, systems that are successfully engineered from a traditional software engineering perspective can be problematic to use or sustain. Part of the problem here is that software and system engineers lack the ways, means, and perhaps ideological perspectives, for observing, experimenting with, and representing the work practices of system users in different places and times [St93, Su95]. To engineers, the work practices of system users are invisible or else are made opaque through formal notations (e.g., source code, object-oriented design notations like UML) and informal graphic diagrams (e.g., flow charts written on white boards). Instead, what if software developers who design the open software-based work systems of users could design more effective systems that increase the performance and amplify the experience of system users? Would such an approach employ *informal or formal representations of user work practices and socio-technical contexts* that were visible and transparent to community participants [BST97,L88,Su95]? If so, what form do such representations take?

C.3.5 Computational modeling of organizational processes and work practices

Research directed at computational modeling and simulation of work practices is a relatively new area of scientific study. Two related approaches are already apparent. First, the work of Scacchi and colleagues [MS90,MS96,S98,S99,S00,SN97,VS99] and others [e.g., CPG98,CS00] demonstrates the ability to model, analyze and simulate complex *organizational processes that are embedded within variety of social conditions, work flows, and technical system configurations.* Second, the work of Clancey and associates, from the former NYNEX Science and Technology and presently at NASA Ames Research Center, focuses attention on the problems of modeling and simulating the *work practices of multiple interacting participants* [CS+96, SiC97]. Clancey and associates report that modeling and simulating work practices differs from conventional approaches to AI and multi-agent systems. In simulating work practices attention is focused on the representation of *how knowledge is situated among work tasks [cf. AM99]*, *how distributed cognition arises in situated settings [cf. AH00]*, and *how work practices act as self-organizing systems*. These concepts raise significant issues for viewing work processes and practices as occurring in specific places, at certain times, within historical circumstances, all as part of *trajectories of work*. Subsequently work practices do not resemble the kinds of models associated with "workflow" systems, nor do work practices readily map onto workflow enactment mechanisms or schemes. Instead, the challenge is to model *how work practices are situated, contingent and continually adapting in incremental, iterative and longitudinal ways* [L88,St93,Su87, Su95]. Subsequently, another emerging line of research has begun to explore how to use Web-based hypertext (i.e., semi-structured) schemes to *describe, situate* (in an information space [MSM95] and workplace), *and explicitly relate data, analysis and computational representations* pertaining to the work structures, processes and practices that arise in settings under study [VS99,S00b].

C.4 Research Approach

There are four primary activities in our proposed research effort. They are presented here in a top-down style. First, we plan to conduct field studies in the four designated open software communities. This is to facilitate comparative analysis in order to increase the generalizability of our results. Second, we plan to identify and case studies of open software work structures, processes and practices. This is also to facilitate comparative analysis in order to increase the generalizability of our results along analytical dimensions. Third, we plan to model and analyze each of the four communities in three ways. Our goal is to model and analyze selected work processes and practices situated within local work structures using (a) computational models, (b) semi-structured (Web-based) descriptions and interlinked artifacts, and (c) narrative descriptions. Fourth, in order to incorporate an ongoing, longitudinal dimension into our study, we plan to iteratively perform each of the other three primary activities each year. Accordingly, these four activities are elaborated in more detail in the following sub-sections.

C.4.1 Comparative Field Studies

We plan to conduct ethnographic field studies of four open software communities using qualitative research methods for collecting, organizing and categorizing our data [GS67]. We will gather data from these communities using participant observation, structured interviews, collection of open software artifacts, and Web-based survey questionnaires. In addition, we will collect secondary materials like bibliographic materials and publications that may describe or acknowledge the development, use or evolution of open software within each community. To explain why and how we intend to proceed, we provide a set of questions and answers to better motivate our choice. These follow.

Why do ethnographic field studies of open software communities? Our research objectives are exploratory and our focus is work-centered. We want to learn how the communities' participants do their work, to what end, why they think it is important, what problems they incur, what changes they desire, and so forth, in the participants own terms [cf. BST97,GS67,L88,St93, Su87]. For example, how do community participants deal with the "free rider" problem described by economists [O71]? Here, a free rider is someone who seeks to benefit from the collective action of community members who build open software as a public good [Sa54], yet are able to possess and use this good without directly contributing some resource, commitment of labor, or reciprocity back to the community. Each of these communities we suspect has many such free riders, yet their presence does not seem to undermine the viability of the community, as would be inferred in classic analyses [O71]. In contrast, if open source code is not the only, nor the most significant, public good at hand, then what might explain the ongoing persistence of such communities? For example, if the primary public good that open software community participants realize is *socio-technical connectivity* and *communality* [MF+98], then these "network externalities" may emerge as a key variable that sustains and evolves these communities. Thus, in order for us to understand and characterize the kinds of work structures, processes and practices that we observe, we will employ ethnographic field studies using qualitative research methods to explore open software communities.

Why do comparative field studies of multiple open software communities? We could choose to study a single, large open software community (e.g., the Linux/GNU community with 1000's of participants). However, we choose to study a comparative set of small communities (each with

10's to 100's of active participants). Variety and comparability rule our choice to study a set of small communities. Similarly, to increase the depth, breadth and scope of our analysis, and to increase the generalizability of our results, we benefit from a choice to study a set of four comparable communities, instead of just one community.

Why study the four designated communities? Our research and programmatic objectives (cf. Section C.2) lead us to explore (a) two academic *research* communities, and (b) two non-academic *development* communities, as our choice for which communities to study. Each community maintains a visible presence of Web-based software, artifacts, documents and other materials. Thus, we have immediate access to these items as data *and* as public points of entry into the community. Nonetheless, we can further describe these four specific communities as follows:

- **Why study the Chandra Science community?** The CS community is based around astrophysical observation and computer facilities at the Harvard-Smithsonian Observatory for Astrophysics, MIT, and TRW in Cambridge, MA [C00]. This is an academic research community whose development, use and evolution of open software for collecting, analyzing and visualizing deep space X-ray observation data is primarily *instrumental* and *practical*. Instrumental use of open software suggests that its development is a means to an end (i.e., develop open software that enables deep space X-ray astrophysics research), rather than a focus on the end itself.
- **Why study the Software Architecture research community?** The SA community is an invisible college of academic Software Engineering researchers working at universities including CMU, Stanford, USC, UC Irvine, Colorado, UT Austin, Imperial College (UK), and elsewhere [cf. SB98,SoAR00]. This is an academic research community whose development, use and evolution of open software tools for modeling, analyzing, simulating and compiling the architectural specification of software systems is primarily an *exploratory, experimental endeavor* in software engineering. Exploratory and experimental development of open software here suggests that such practice is both a means to an end (developing and using these software tools enables this line of research work), and the end itself (building software tools/artifacts is interesting and professionally rewarding to academic software experts).
- **Why study the Apache Project community?** Apache (www.apache.org) is a globally distributed software development project coordinated by a group of some 20 developers [F99]. This is a non-academic development community (with some exceptions [F99]) whose development, use and evolution of open software for the world's most popular Web servers is *instrumental* and *practical* to most of its participants. However, it is also *an exploratory, experimental endeavor* in globally distributed software engineering to other participants [C99, FW+98,K99,NS99].
- **Why study the Unreal computer game community?** Unreal is a computer action game (i.e., a "first person shooter") whose conceptual lineage spans earlier games like *Quake, Heretic* and *Doom*, with another 20 new computer games as successors licensed to use its engine. It is an open, extensible game environment built about a closed, proprietary game engine developed by the firm, Epic Games Inc. [U00]. Nonetheless, it has many 100,000's users and at least 100's of public developers and user groups who share game extension software or content (e.g., textured 3D geometric models) through both a formally orchestrated [cf. K00] and informal community of Web sites and network servers. It

represents a non-academic development community (dominated by young males, 12-30 years old) whose development, use and evolution of open software for extending game play and game content for a proprietary game engine is primarily *instrumental, practical* and a *source of personal enrichment* (e.g. having fun playing new game variations).

Are these the only four open software communities to consider? No, but as a set, they do display an interesting and contrasting set of characteristics. Also, we note two dimensions for comparing the communities. First is the grouping of two academic, research-oriented communities, and two non-academic, development-oriented communities. Second, the grouping of two communities whose interest in open software is primarily instrumental, and two whose interests are exploratory and experimental. Thus, with these two dimensions we have an initial basis for comparative analysis across these designated communities. Beyond this, other plausible open software communities that merit consideration for systematic study include the Internet Engineering Task Force (IETF) [cf., DOS99], World-Wide Web Consortium (W3C), the National Institute of Standards and Technology (NIST), the Internet privacy working group (IPWG/P3P), and the Linux/GNU [T99,St99,DOS99] operating system environment community. Why not study one of these? For IETF, we bring no prior experience or legacy of domain competency. Thus getting familiar and knowledgeable about the domain would be time-consuming and slow the research effort. The W3C and NIST primarily focus their efforts on the development of software standards, rather than the development or evolution of open software. Thus, they represent different kinds of communities that we are addressing. Similarly, the P3P community is active in developing open privacy guidelines within the W3C community. Ackerman is an active participant in this community, so access is in hand. However, the choice to include P3P in this study would require modification of the comparative community research design described earlier. Finally, the Linux/GNU community is now very large, and increasingly populated with commercial enterprises, start-up firms backed by venture capital, and the like. This community perhaps thus merits a separate study on its own, since it should be possible to explore both the social and economic implications of the Linux/GNU operating system environment.

How do we intend to collect field study and case study data? We will collect data through participant observation, semi-structured interviews, artifact gathering, and Web-based questionnaires. Participant observation of these communities may be direct (e.g., attending work group meetings or practitioner conferences) and indirect (corresponding through email with participants; follow distribution lists, examine Web sites [MA96]). This method requires the research team members to first map out what they know about each community, and what community-based resources or artifacts can be located on/off the Web. Following this, the team can identify initial people to contact for interview in each community. For each of the four communities, we can identify and locate (via email) these people. Follow-on interviews with other community participants will then be determined through interaction and semi-structured interviews with initial contacts, as well as through the initial coding and review of this data [cf. GS67]. Our goal is to interview a total of 40-60 participants across the four communities each year, at a minimum. Beyond this, Web-based questionnaires will be employed to solicit input or feedback on emerging models from dispersed community participants. Note we are not claiming that these surveys will follow an experimental research design intended to produce robust data for extensive statistical or quantitative analysis. Instead, we will use convenience surveys as indicators for remote community participants to voluntarily contribute their views on questions that we will ask, which follow from initial analysis of our other primary data sources. These weak indicator data may nonetheless shed light on the frequency and distribution (spatial, temporal, or geographic) of events, situations, or conditions we discover from interviews and observations. Similarly, these weak indicators may enable us to conduct *crosscutting, triangulation analyses* that begins to reconcile qualitative and quantitative data in moving toward the development of grounded theory [GS67,Y89].

How can we identify representative demographics of each community under study? A community can be characterized in a number of ways. These include its *membership* (who's in, who's not), extent of partially overlapping *networks of relationships, mutual commitmen*t and *generalized reciprocity* among participants, an affinity of *shared values* and *practices* (including shared experiences, histories, norm and incentive structures, processes, and problematic situations), collective creation and distribution of *collective goods,* and *persistence* of the community [H55]. Similarly, an open software community can be characterized in terms of the number and geographic distribution of participants, frequency and kind of participation they engage, community events that trigger differentiated participation, social processes or mechanisms for integrating or discouraging the inputs from prospective new participants, etc. We intend to use "free" questionnaire software/services readily available on the Web (e.g., www.zoomerang.com) to collect data such as these. Using these software/services it is possible to rapidly create and deploy Web-based questionnaires that can collect and analyze (via descriptive statistics) data as it is submitted. Our strategy here is to use this kind of questionnaire (using privacy guards like login/user-id pseudonyms as personal identifiers) to conduct convenience surveys targeted to members in a community. Acknowledged sponsorship from NSF, as well as affinities established by our research team through participant observation in face-to-face meetings and via email postings, will be employed to encourage Web-based questionnaire completion, submission and feedback. In this regard, we will make the quantitative results of our questionnaire open to community members, and subsequently observe whether such feedback induces any changes in the community, as well as the form it takes. [cf. OS00].

How can we gain access to participants and artifacts in each community? Our research team has established access with each of the four communities. Ackerman has access and periodic interaction with participants in the Chandra Science community (see for example [AM99]). Scacchi has access and periodic interaction with participants in the Software Architecture research community through prior work [e.g., SB98] and through his participation at the UC Irvine Institute for Software Research. He also accesses and engages the Unreal game community through his NSF funded project on Enterprise Visualization. Ackerman has also conducted field study in a computer game community [MA96]. Last, UCI ICS graduate students [e.g., F99, FW+98] among others are participants in the Apache Project, and our research assistants will have access and periodic interaction with them.

How can we collect and analyze open software artifacts from each community? Participants in each of these communities make routine and extensive use of the Web to mediate their work, and to store/share the products and artifacts of their work. Thus, we intend to characterize and document these community resources and public artifacts as part of the data we seek to collect, analyze, model and archive. This requires us to develop an inventory of the types and descriptive forms used to collect and share information about software programs, artifacts, test data, etc. within each community.

How can we identify the open software work structures, processes and practices that are central to collective action in an open software community? To start, we prefer a broad view of what work structures can be. In our view, work structures are not pre-defined forms or patterns that exist independent of software development work. Instead, we prefer to view work structures as a continually emerging composition (or recomposition [G98]) of teamwork relations, patterns of work flow, resources arrangements that situate work activities and constitute workplaces, as well as the larger institutional contexts in which communities participate and operate. Work structures are thus a way of viewing how software development workplace settings and resource arrangements facilitate and constrain who does what, where, when, how and why, as well as with whom and with what.

Next, open software work processes denote *classes* or *patterns of recurring work practices* in a setting [MS90,MS96,SM97]. Such processes may recur within a community in one/many spatio-temporal places or dispersed cognitive-computational information spaces [cf. CoB99,MCM95, Sh91]. Similarly, they may recur across more than one community. Processes differ from practices at an analytical level of abstraction, in that work practices can be viewed as *instances* of either recurring or emerging work processes, or as individual/collective actions (including social discourse) toward situated events [P98]. This allows for work practices that may represent *unique* or *non-recurring processes*, which then may be characterized or modeled as such [cf. SM97,CS+96,SiC97]. From a traditional software engineering perspective [So95] such non-recurring processes are not the subject of attention, nor a basis for providing automated workflow or process enactment support. Nonetheless, in our study, we seek to identify and characterize *open software work structures, processes and practices as our units of analysis*, as well as to the participants, artifacts, resources, information infrastructures, historical circumstances and institutional contexts that situate them, within and across four communities of practice.

C.4.2 Comparative Case Studies

Through our discussion of related research in Section C.3, we identified five analytical categories and highlighted a number of analytical themes we will use as lines of inquiry in our field studies. These categories and themes are combined and condensed into the units of analysis, as well as the units for comparison, in our study. Subsequently, a given exploration of one or more unit of analysis situated within or across an open software community constitutes a case study in our research approach [Y98]. For example, if we examine the processes and practices by which the Chandra Science community modifies and repairs its open software (programs, artifacts, documentation, email postings) for astrophysical data collection and visualization, this would constitute a case study. Alternatively, if we address how the Unreal computer game community disperses new "game levels" throughout its community of practice (via Unreal Web sites, game and list servers), this would also denote a case study. Similarly, if we address how either the Software Architecture community or Apache Project practices software engineering design and testing methods (e.g., from textbook references, research publications, or ad hoc locally grown sources) to realize what they consider reliable software, these too would be case studies. Consequently, we then see that re-examining the units of analysis of a given case study in one or more other open software communities would enable us to compare and contrast such units through multiple comparative case studies. This means we can examine, compare and contrast how each of the four communities, for example, (a) modifies and repairs its open software, (b) communicates and disperses new open software components, or (c) designs and tests open software systems in order to determine (or "define") their reliability.

Overall, we seek to conduct comparative case studies that identify and characterize *common* open software work structures, processes, practices and artifacts arising (a) within a community, (b) across communities, and (c) across analytical categories or empirically driven theoretical samples [GS67], as we and other researchers have done before [e.g., BS89,GHP99,JS95, KS82, S98]. Similarly, we seek to identify and characterize *distinctive* open software work structures, processes, practices and artifacts arising (a) within a community, (b) across communities, and (c) across analytical categories or empirically driven theoretical samples. Subsequently, our choice for which analytical categories or theme to employ in one or more case studies will emerge from a mixture of the initial maps of each community we develop, the data we sample and collect from participant observation and structured interviews with people in each community, and through individual/professional subject-matter interests of our research team members.

C.4.3 Modeling and Analysis

Each case study provides data for modeling and analysis. We anticipate developing three kinds of models that represent the socio-technical work structures, processes and practices of open software communities. Each kind of model employs a different representational medium or notational form, though all such models can be published for access over the Web.

First, we plan to develop *computational models* and analyses that provide a formal representation of selected open software work structures, processes or practices in a form that can be parsed, analyzed or interpreted with computer-based tools [cf. MS90,MS96,SM97,S99].

Second, we plan to develop *semi-structured models* and analyses that allow us to explicitly interrelate online case study data, artifacts, annotations and analyses to computational models and descriptive narratives using knowledge-based modeling [MS96], Web-based mechanisms [SN97] and hypertext techniques [NS99,VS99,S00b]. What makes these models semi-structured is their combination of multiple heterogeneous notations, descriptions and computational models of work structures, processes or practices that are cross-linked (or "hyperlinked") for exploratory browsing and comparative analyses [cf. GS67,MSM95,Sh91].

Third, we plan to develop *narrative models* that articulate a story grounded in situated discourse data to characterize, explain, or otherwise provide the participants' interpretation of the focal work structures, processes, practices or other embedding situation(s) at hand [cf. AH00,BS89, JS95,KS82,McA98,MA98,Su87]. The associated analyses will articulate, compare and contrast alternative explanations and interpretations that are drawn from other case studies or from the research literature.

Fourth, given the preceding three classes of models for describing and representing case study data, we can then address issues pertaining to the comparative ability for one class of models to more/less comprehensively portray the underlying socio-technical phenomena being modeled and analyzed. This we believe is an important new step in addressing concerns that have been raised in the literature regarding the relative efficacy of different narrative vs. computational schemes for representing (or occluding) socio-technical work structures, processes or practices [cf. Su95].

Finally, our task is to generalize from the previous sets of empirical grounded models in order to develop, articulate and refine a *meta-model* of open software work structures, processes and practices [MS96]. This is most easily done working from computational models [MS90,S99, SM97]. Nonetheless, such a meta-model constitutes a kind of empirically grounded *theory* of open software work structures, processes and practices that we seek to develop through this research study.

C.4.4 Multiple Iterations for Longitudinal Analysis

We plan to engage, observe, collect and analyze data from each community for a three-year project duration. We also plan to iterate study of each community each year, hence three iterations. This means plan to conduct field studies each year that will focus on identifying, exploring and revisiting locally significant work structures, processes or practices as case studies that can be compared within or across field sites, over time. Similarly, it also suggests the need to go back and re-interview participants in order to elicit their views on events or changing conditions that have appeared in the interim. Beyond this, we plan to model and analyze the work structures, processes and practices that correspond with those of each case study. In turn, this may lead to the iterative modeling, comparative analysis and incremental refinement and evolution of cases within or across the three study years. Collectively, these will enable us to conduct a longitudinal study of open software communities, processes and practices in an iterative, incremental and comparative manner.

C.4.5 Project staffing and resources

Our proposed research team consists of two senior research investigators (Ackerman and Scacchi), assisted by two graduate research assistants. Both investigators have a history of prior experience and publications in research projects that primarily employ ethnographic methods as a basis for developing systematic understanding and/or empirically grounded theory. With two investigators and two research assistants, we have the requisite staff needed to undertake exploratory investigations, grounded theory research methods, data collection, analysis and modeling in the four designated communities being studied. Each research team member will be primarily responsible for leading the research effort in one community. Each team member will provide research support (secondary data collection, review of collected artifacts, analyses and models) from two other communities.

Based on prior experience, each project team member can interview, code and review data from 6 to 10 community participants in a person-week of effort (though such effort is not usually so contiguous). Browsing, collecting, analyzing and interlinking open software artifacts, documents or related bibliographic materials from each community can be done on an ongoing basis throughout the project. Modeling, analysis and write-up of the socio-technical work structures, processes and practices characteristic of each open software community is estimated at two-to-four person weeks of effort, per model. Our goal is to produce four to eight models in narrative, semi-structured and computational forms (i.e., 12 to 24 models) per project year. These models will then be revised and refined through ongoing data collection and analysis, so 20-30 models (or "cases") may ultimately be produced after three years of research effort, though half that number would still constitute a substantial base of results.

C.5 Anticipated Research Outcomes

We anticipate six kinds of research outcomes will result from our proposed research effort. These are identified in turn.

C.5.1 Models of Open Software Communities, Work Structures, Processes and Practices

We will develop computational, semi-structured and narrative models that describe socio-technical work structures, processes and practices associated with the production and evolution of open software, as well as the communities of practice from which they emerge.

C.5.2 Develop Foundations for Empirically-Grounded Theory

We ultimately seek an empirically grounded theory that can characterize, explain and predict (within limits) the socio-technical work structures, processes and practices that enable the production and evolution of open software within the kinds of open software communities that we study. Such theory must both draw on the various types of models we will develop, as well as the underlying data and artifacts. Subsequently, the theory must serve as a meta-model that covers the individual models [cf. MS96]. Additionally, others researchers not directly involved in this project must be able to independently review, re-analyze and assess our theory to determine whether alternative models or interpretations will contradict or refute those we have made. Thus, our models and analyses will be published for remote access on the Web.

C.5.3 Dissemination of Selected Data

We plan to publish selected data and secondary materials that provide the base for our model building and theory development, while maintaining the confidentiality and privacy of participants interviewed during the field studies. This data can also serve as a resource for others to employ to re-analyze, extend or refute our efforts, as the practice of grounded theory and comparative case study encourages [cf. GS67,Y89].

C.5.4 Research Publications and Presentations

We plan to publish and present our models, theory and data in journal articles and international conferences, symposia or workshops addressing audiences in the areas of social and organizational informatics, computer-supported cooperative work environments, software engineering, software development practice and experience, organizational and software processes.

C.5.5 Research, Education and Community Interaction

The anticipated products from this research noted above will be integrated into graduate and later undergraduate coursework at UCI, particularly in courses addressing software engineering, computer-supported cooperative work, and social/economic implications of computing. In addition, the models and publications we will develop are likely to be of interest to the corporate affiliates of the UCI Institute of Software Research. Thus, we have the opportunity and the situated audiences to both present and elicit feedback from regarding the emerging models and results we develop during the period of this project.

C.5.6 Guidelines for Developing an Open Software Community

If our research effort is successful in meeting all of its research and programmatic objectives, it should be possible to begin to articulate the practices and lessons that open software community participants have learned. To be clear, emphasis here is on identifying the practices and lessons learned in the terms that the participants assert are useful, rather than having us, the research team, making such judgement on what works best. Our role is not to make such judgements, rather our purpose is to help elicit, document, compare and share those from the participants in different open software communities. Thus, our commitment here to produce such guidelines is tentative, though there are plausible reasons for expecting that such advice will be sought by readers and reviewers of our research results and publications.

C.6 Results from Prior NSF Supported Research

Mark S. Ackerman's (Co-PI) CAREER grant **IRI-9702904** examines the organization of networks of expertise, including ad-hoc networks. As part of this work, he conducted the analysis of organizational memory [AH00]. In addition, a dissertation under his supervision has examined how people seek expertise within a software company, uncovers mechanisms by which expertise is found and transmitted in software environments [McA98].

Walt Scacchi (PI) has had no prior support from NSF since his days in graduate school in 1974-1980. Beginning January 2000, Scacchi is funded as a Senior Scientist from the NSF SBIR Program on a grant **DMI-9960830** titled "SBIR Phase I: Enterprise Visualization" for the period January-June 2000 to Bayfront Technologies Inc. in Costa Mesa, CA. This grant supports Scacchi at 50% time. There have been no results published from this effort to date, though the project is on schedule, and results will be documented and published in spring 2000. Otherwise, Scacchi has been PI or Co-PI on 25 externally funded research contracts or grants from various government agencies and commercial firms.

D. References

[**AH00**] M.S. Ackerman and C. Halverson. Re-examining an Organization's Memory. *Communications ACM,* 43(1):58-64, January 2000.
[**AM99**] M.S. Ackerman and E. Mandel. Memory in the Small: Combining Collective Memory and Task Support for a Scientific Community. *J. Organizational Computing and Electronic Commerce*, 9(2-3):105-127, 1999.
[**B00**] K. Beck. *Extreme Programming Explained*, Addison Wesley Longman, Reading, MA 2000.
[**BS89**] S. Bendifallah and W. Scacchi. Work Shifts and Structures: An Empirical Study of Software Specification Work, *11th. Intern. Conf. Software Engineering*, Pittsburgh, PA, ACM Press, 260-270, May 1989.
[**BHP89**] W.E. Bijker, T.P. Hughes and T.F. Pinch. *The Social Construction of Technological Systems : New Directions in the Sociology and History of Technology*. MIT Press, 1989.
[**BD91**] J.S. Brown and P. Duguid. Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. *Organization Science*, 2(1):40-57, 1991.
[**BS96**] G. Button and W. Sharrock. Project Work: The Organisation of Collaborative Design and Development in Software Engineering. *Computer Supported Cooperative Work*, 5(4): 369-386, 1996.
[**BST97**] G. Bowker, S.L. Star, E. and W. Turner. *Social Science, Technical Systems, and Cooperative Work : Beyond the Great Divide*, Lawrence Erlbaum, 1997.
[**CPG98**] K. Carley, L. Gasser, and M. Prietula (eds.), *Simulating Organizations: Computational Models of Institutions and Groups*, MIT Press, 1998.
[**C99**] E. Carmel. *Global Software Teams: Collaborating Across Borders and Time Zones*. Prentice-Hall, 1999.
[**C00**] Chandra Science Web Site, http://asc.harvard.edu/. Chandra Software Tools Information Site: http://asc.harvard.edu/udocs/docs/docs.html, January 2000.
[**CS00**] A.M. Christie and M.J. Staley. Organizational and Social Simulation of a Software Requirements Development Process, *Software Process--Improvement and Practice*, (to appear), 2000.
[**CS+96**] W.J. Clancey, P. Sachs, M. Sierhuis and R. van Hoof, Brahms: Simulating Practice for Work Systems Design, Invited presentation at Pacific Knowledge Acquisition Workshop, Sydney, October 1996.
[**Co68**] M.E. Conway. How do Committees Invent?, *Datamation*, 14(4):28-31, 1968.
[**CoB99**] S.D. Cook and J.S. Brown. Bridging Epistemologies: The Generative Dance between Organizational Knowledge and Organizational Knowing. *Organization Science*, 10(4):381-400, July 1999.
[**DOS99**] C. DiBona, S. Ockman and M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*. O'Reilly & Associates Inc., Sebastol, CA 1999.
[**F99**] R.T. Fielding. Shared Leadership in the Apache Project. *Communications ACM*, 42(4):42-43, 1999.
[**FW+98**] R.T. Fielding, E.J. Whitehead, K.M. Anderson, G.A. Bolcer, P. Oreizy and R.N. Taylor, Web-Based Design of Complex Information Products. *Communications ACM*, 41(8):84-92, 1998.

[**GIK95**] J.F. George, S. Iacono and R. Kling. Learning in Context: Extensively Computerized Work Groups as Communities-of-Practice. *Accounting, Management and Information Technology*. 5(3/4):185-202, 1995.
[**GS67**] B. Glaser and A.L. Strauss. *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Aldine, New York, 1967.
[**G96**] R. Grinter. Supporting Articulation Work Using Software Configuration Management Systems. *Computer Supported Cooperative Work*, 5(4): 447-465, 1996.
[**G98**]R.E. Grinter. Recomposition: Putting it All Back Together Again. *Proc. ACM Conference Computer Supported Cooperative Work (CSCW '98)*. Seattle, Washington: November 14-18. 393-403.
[**GHP99**] R. Grinter, J.D. Herbsleb and D.E. Perry. The Geography of Coordination: Dealing with Distance in R&D Work. *Proc. ACM SIGGROUP Conf. on Supporting Group Work*, Phoenix, AZ, 306-315, November 1999.
[**H55**] G.A. Hillery. Definitions of Community: Areas of Agreement. *Rural Sociology*, 20:111-123. 1955.
[**ID98**] A.C. Inkpen and A. Dinur. Knowledge Management Processes and International Joint Ventures. *Organization Science*, 9(4):454-468, July 1998.
[**JS95**] A. Jazzar and W. Scacchi. Understanding the Requirements for Information Systems Documentation, *Proc. 1995 ACM Conf. Organizational Computing Systems*, San Jose, CA, ACM Press, 268-279, August 1995.
[**K99**] D.W. Karolak. *Global Software Development: Managing Virtual Teams and Environments*. IEEE Computer Society Press, Los Alamitos, CA 1999.
[**K00**] A.J. Kim. *Community Building on the Web: Secret Strategies for Successful Online Communities*, PeachPit Press, 2000.
[**K94**] R. Kling. Cooperation, Coordination and Control in Computer-Supported Work. *Communications ACM*, 34(12):83-88, December 1994.
[**KS82**] R. Kling and W. Scacchi, The Web of Computing: Computer Technology as Social Organization, in A. Yovits (ed.), *Advances in Computers*, 21, Academic Press, 3-85, 1982.
[**K-C99**] K.Knorr-Cetina. *Epistemic Cultures: How the Sciences Make Knowledge*. Harvard University Press, Cambridge, MA 1999.
[**KoL99**] M.P. Koza and A.Y. Lewin. The Coevolution of Network Alliances: A Longitudinal Analysis of an International Professional Service Network. *Organization Science*, 10(5):638-653, 1999.
[**KrS95**] R.E. Kraut and L.A. Streeter. Coordination in Software Development. *Communications ACM*. 38(3):69-81. March 1995.
[**L88**] B. Latour. *Science in Action : How to Follow Scientists and Engineers Through Society*, Harvard University Press, Cambridge, MA, 1988.
[**LW91**] J. Lave and E. Wegner. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press, Cambridge, 1991.
[**MSM95**] C.C. Marshall, F.M. Shipman and R.J. McCall. Making Large-Scale Information Resources Serve Communities of Practice. *J. Management Information Systems*. 11(4):65-86, 1995.
[**MO93**] G. Marwell and P. Oliver. *The Critical Mass in Collective Action : A Micro-Social Theory*. Cambridge University Press, 1993.

[**McA98**] D.W. McDonald and M.S. Ackerman. Just Talk to Me: A Field Study of Expertise Location, *Proc. ACM Conf. Computer Supported Cooperative Work (CSCW'98),* Seattle, WA, ACM Press, 315-324, 1998.
[**MA98**] J. Muramatsu and M.S. Ackerman. Computing, Social Activity and Entertainment: A Field Study of a Game MUD. *J. Computer Supported Cooperative Work*, 7(1):87-122, 1998.
[**McK99**] M.K. McKusick. Twenty Years of Berkeley Unix: From AT&T-Owned to Freely Distributable, in [DOS99], 31-46, 1999.
[**MF+98**] P.R. Monge, J. Fulk, M.E. Kalman, A.J. Flanagin, C. Parnassa and S. Rumsey. Production of Collective Action in Alliance-Based Interorganizational Communication and Information Systems. *Organization Science*, 9(3): 411-433, 1998.
[**MS90**] P. Mi and W. Scacchi, A Knowledge-Based Environment for Modeling and Simulating Software Engineering Processes, *IEEE Trans. Data and Knowledge Engineering*, 2(3): 283-294, September 1990.
[**MS93**] P. Mi and W. Scacchi, Articulation: An Integrative Approach to Diagnosis, Replanning and Rescheduling, *Proc. 8th. Knowledge-Based Software Engineering Conf.*, Chicago, IL, 77-85, 1993.
[**MS96**] P. Mi and W. Scacchi, A Meta-Model for Formulating Knowledge-Based Models of Software Development, *Decision Support Systems*, 17(4): 313-330, 1996.
[**NS87**] K. Narayanaswamy and W. Scacchi. Maintaining the Configuration of Evolving Software Systems, *IEEE Trans. Software Engineering*, 13(3):324-334, March, 1987.
[**NS97**] J. Noll and W. Scacchi. Supporting Distributed Configuration Management in Virtual Enterprises. in R. Conradi (ed.), *Software Configuration Management*, Lecture Notes in Computer Science, Vol. 1235, Springer-Verlag, New York, pp. 142-160, (1997).
[**NS99**] J. Noll and W. Scacchi. Supporting Software Development in Virtual Enterprises. *J. Digital Information*, 1(4), February 1999. http://journals.ecs.soton.ac.uk/jodi/
[**O71**] M. Olson. *The Logic of Collective Action*, Harvard University Press, Cambridge, MA, 1971.
[**OS00**] The Open Science Project Web Site, http://www.openscience.org, January 2000.
[**P98**] M.F. Peterson. Embedded Organizational Events: The Units of Process in Organization Science. *Organization Science*, 9(1):16-33, 1998.
[**Sa54**] P. Samuelson. The Pure Theory of Public Expenditure. *Review of Economics and Statistics*, 36:387-390, 1954.
[**S84**] W. Scacchi. Managing Software Engineering Projects: A Social Analysis, *IEEE Trans. Software Engineering*, 10(1):49-59, January 1984.
[S98] W. Scacchi, Modeling, Integrating and Enacting Complex Organizational Processes, in [CPG98], 153-168, 1998.
[**S99**] W. Scacchi, Experience with Software Process Simulation and Modeling, *J. Systems and Software*, 46(2-3): 183-192, 1999.
[**S00**] W. Scacchi. Redesigning Contracted Service Procurement for Internet-based Electronic Commerce: A Case Study. *J. Information Technology and Management,* (to appear), 2000.
[**S00b**] W. Scacchi. Understanding Software Process Redesign using Modeling, Analysis and Simulation. *Software Process--Improvement and Practice*, (to appear), 2000.
[**SB98**] W. Scacchi and B.E. Boehm. Virtual Systems Acquisition: Approach and Transitions. *Acquisition Review Quarterly*, 5(2):185-216, 1998.
[**SN97**] W. Scacchi and J. Noll, Process-Driven Intranets: Life Cycle Support for Process Reengineering, *IEEE Internet Computing*, 1(5):42-49, 1997.

[**SM97**] W, Scacchi and P. Mi. Process Life Cycle Engineering: Approach and Environment. *Intelligent Systems in. Accounting, Finance, and Management*, 6:83-107, 1997.
[**Sh92**] B.R. Shatz. Building an Electronic Community System. *J. Management Information Systems*, 8(3):87-107, 1992.
[**SiC97**] M. Sierhuis and W.J. Clancey, Knowledge, Practice, Activities and People, Paper presented at the AAAI Spring Symposium on AI in Knowledge Management, 1997.
[**SiMG99**] C. Simone, G. Mark, and D. Giubbilei. Interoperability as a Means of Articulation Work. *Proc. Intern. Conf. Work Activities Coordination and Collaboration (WACC'99),* San Francisco, ACM Press, 39-48, February 1999.
[**SoAR00**] Software Architecture Research Web site, http://www.ics.uci.edu/pub/arch/. Also see http://www.isr.uci.edu/events/wesas2000/.
[**So95**] I. Somerville. *Software Engineering* (Fifth Edition), Addison Wesley Longman, Reading, MA 1995.
[**St99**] R. Stallman. The GNU Operating System and the Free Software Movement. in [DOS99], 53-72, 1999.
[**SR96**] S.L. Star and K. Ruhleder. Steps Toward an Ecology of Infrastructure: Design and Access for Large Information Spaces. *Information Systems Research*, 7(1):111-134, March 1996.
[**St93**] A. Strauss, *Continual Permutations of Action*, Adeline De Gruyter, New York, 1993.
[**Su87**] L.A. Suchman. *Plans and Situated Actions: The Problem of Human-Machine Communication*, Cambridge University Press, 1987.
[**Su95**] L. Suchman, Making Work Visible, *Communications ACM*, 38(9):56-64, 1995.
[**T99**] L. Torvalds. The Linux Edge. In [DOS99], 101-112, 1999.
[**U00**] Unreal World Web Sites: http://www.unreal.com, http://unreal.epicgames.com, http://www.unrealized.com, http://www.unrealty.org, http://www.unrealengine.com, (plus 100 other sites), January 2000.
[**VS99**] A. Valente and W. Scacchi. Developing a Knowledge Web for Business Process Redesign. *Twelve Workshop on Knowledge Acquisition Modeling and Management*, Banff, Canada, October 1999. http://sern.ucalgary.ca/KSI/KAW/KAW99/papers.html
[**W98**] E.Wenger. *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press, 1998.
[**Y89**] R.K.Yin. *Case Study Research: Design and Methods*. Sage Publishers Inc, Newbury Park, CA, 1989.