SAND2011-3148C

Interaction Design for Visualization of Large-Scale Data Sets

William A. Stubblefield[1]
Sandia National Laboratories

Laura A. Mcnamara[2]
Sandia National Laboratories

Jason Shepherd[3]
Sandia National Laboratories

ABSTRACT

Tools for visualizing and analyzing large data sets too often fail to secure adoption by their intended users. Usability problems are often a factor, but a deeper cause is a misunderstanding of the social and physical context of data analysis and visualization. A powerful tool for understanding how people use large data can be found in activity theory and other models of how tools and information mediate collaborative work. We demonstrate its power through an empirical study of a prototype analysis tool that failed to secure adoption, and argue for the use of activity theory in designing tools to support large data set analysis.

KEYWORDS Interaction design, activity theory, data visualization

1. INTRODUCTION

As ultra-large data sets become increasingly important to science [1], scientists become increasingly dependent on visualization and analysis tools to interpret that data [2]. Paradoxically, as these tools grow in sophistication and potential, securing user adoption remains difficult. We believe that the reason so many powerful data analysis tools struggle to find adoption is a failure to afford users with visual representations and interactive behaviors that engage the complex, adaptive, deeply nuanced patterns of activity that scientific and other knowledge creation communities require for their vitality. In this paper, we propose the use of activity theory as a framework for understanding the ways people use large data sets, and improving the design of tools to support them.

Rapid changes in the size, availability, and importance of large-scale data creates tremendous challenges for data analysis and visualization, including performance limitations of existing hardware and software architectures; matching visual vocabularies to target problems; coordinating analysis and visualization across distributed datasets; managing data and visualization provenance information; and keeping visualization environments up-to-date as datasets evolve both qualitatively and quantitatively [3]. In addition, scientific visualization confronts discipline-specific challenges: for example, mathematical modeling in systems biology involves iterative exploration of multiple solution spaces, creating "landscapes of local optimalities" that are difficult to represent comprehensively [4]. These technical complexities are reflected in the complexity of the communities that conduct data-intensive science. This, and the rapid progress of visualization research challenge our ability to turn new technologies into useful tools, and users' ability to integrate them into their work.

Given human reliance on visual information processing, visualization software is critical in enabling researchers to interact productively with massively large and complicated data. Humans lack the perceptual and cognitive abilities to make sense of large data sets without machine assistance, and scientific disciplines recognize that visualization tools are likely to play an important role in future work. Yet despite sustained attention to the technical challenges of large-scale data visualization and widespread interest in incorporating visualizations into research, getting researchers to adopt novel visualization tools can be challenging. In spite of designers' efforts to address user needs, responses can be surprisingly lukewarm. The authors know of several instances in which scientific and engineering users have eschewed novel visualization software in favor of analytical tools that generate simpler, more conventional representations of information – such as scatterplots – even as they acknowledge the limitations of these representations for complex analytic problems.

We believe that many, if not most, adoption failures can be attributed to inadequate consideration of the social, physical, and organizational contexts of work. Because these communities seek to create new knowledge from large, as yet unanalyzed data sets, they not only involve collaborations across diverse disciplines, but also reconfigure their work practices continuously to meet the emerging demands of exploratory data use. To address these problems, designers must shift their focus from the ways that visualization and analysis tools search and transform data, to characterizing the ways in which they mediate the complex technical, social, and cognitive dance of exploratory science.

In the following pages, we present activity theory as a useful way to frame the complex mediations among visualization researchers, visualization software and hardware, datasets, and the multidisciplinary communities for whom these tools are intended. Activity theory treats human cognition as goal-directed practice that is inextricably embedded in a "social matrix composed of people and artifacts," as well as objects, outcomes, norms, roles, and meanings [5]. We briefly describe the philosophical and historical roots of activity theory, sketch its grounding concepts, and identify some approaches suggested by an activity theoretic orientation to design. However, the bulk of this paper is a case study in which we use activity theory to explain why an analytic user community did *not* adopt a prototype text analysis visualization tool. This example not only calls our attention to contextual factors that designers must consider, but also can help visualization developers align their research goals with those of the target user community, and improve their ability to create usable, useful, and adoptable large-scale data visualization and analysis technologies.

2. VISUALIZATION DESIGN

The idea that technologies should address the needs of their users is hardly revolutionary: the field of human factors emerged during World War II to help aircraft designers reduce human error. In the

[1]email: wastubb@sandia.gov
[2]email: lamcnam@sandia.gov
[3]email: jfsheph@sandia.gov

1980s, human factors and computer science intersected, establishing the field of human-computer interaction (HCI), and the HCI community expanded rapidly as computers became smaller, more affordable, and increasingly ubiquitous. Today, HCI is a diverse field of study, spanning disciplines, from cultural anthropology to computer science.

Interestingly, several of the leading figures in early HCI movement, including Ben Shneiderman and Stuart Card, also played foundational roles in establishing information visualization and visual analytics as computer science subdisciplines. This is not surprising, given that information visualization is inherently aimed at augmenting human abilities to interpret complex information [6]. Computer-supported visualization research has had significant societal impacts, as novel visual vocabularies and software tools have migrated into the mainstream. For example, financial journalists have adopted interactive tree maps to render market data in comprehensible form [7, 8], while relatively unskilled users can access on-line visualization environments and data to create sophisticated information representations [9]. This spread of large data visualization tools into diverse domains further emphasizes the importance of user-oriented design methods.

Yet even with these successes, many visualization experts believe the field has not realized its potential. As Shneiderman et al. observed, visualization researchers "envision profound impacts and widespread application, but the reality is often more sobering" [10]. As many information visualization and visual analytics researchers have lamented, designing visualization environments to meet the analytic needs of a diverse user groups is difficult, even when datasets are relatively small. Equally difficult is evaluating the extent to which visual analytics environments enhance the usefulness of information. Many visualization projects struggle with issues of usability, utility, adoptability, and technology commercialization, even as they are generating novel and exciting techniques for interacting with information.

Perhaps because of its inherent orientation toward perception and cognitive processing, visualization research has long drawn upon psychological models of attention, memory, and perception [6]. Even so, the "field of information visualization offers little practical guidance to practitioners who seek to design novel systems" [11]. This is despite the ubiquity of frameworks like Shneiderman's theory of visual information seeking [12], which is perhaps the most widely cited design guidance in the visualization literature. To address these challenges, visualization researchers have recently started looking beyond cognitive science for other paradigms, methods, and theories to conceptualize the relationship between human activity and visualization technologies, and to support visualization tool design and evaluation [13-16].

The challenges of designing data analysis software extend beyond the immediate user-machine relationship in ways that theories of individual cognitive processing cannot address. As people use computers to perform work, they engage in complex reciprocal exchanges with other people, computer software, and computer hardware. Moreover, places of work are situated within broader organizational, financial, social, and historical contexts. Not only do these broader contexts shape the meaning and practice of work at the local level; but also in reciprocal fashion, human activity evolves these higher-order dynamics. Indeed, activity theory and other models of socially situated cognition took shape in the early 1990s as HCI practitioners recognized the limitations of cognitive science-based approaches to human-computer interaction [17].

These issues are particularly salient in the design of large-scale data analysis and visualization tools, because data-intensive science increasingly requires large, multi-disciplinary collaborations, as are now the norm in fields like climate science and economics. Traditional software requirements and design methods assume stable, complete, and unambiguous specifications of technology, information, work processes, and social organization that scientific research communities cannot provide. Developing tools that help these communities to interact meaningfully with their data and with each other, requires careful attention to the web of actors and artifacts that comprise exploratory research. The next section introduces ideas of interaction design, artifact-mediated collaboration, and activity theory as tools to help visualization researchers develop useful, usable and adoptable technologies that meet the rapidly evolving demands presented by massive data.

3. Activity Theory and its Foundations

The central idea of interaction design [18, 19] is that we must think of the tools, interactive behaviors, organizational structures, work processes, infrastructure, and other factors surrounding a community of practice as a single, interconnected system. Interaction design provides a number of methods and theories for doing this, including activity theory. However, using these techniques requires a fundamental shift in thinking about the nature of computation, interactivity, and the relationship between tools, information, people, and culture. This section addresses that shift by reviewing the intellectual foundations of interaction design.

As computer scientists, we are trained to think of information and information processing in formal terms: as algorithms operating on well-formed syntactic expressions. Under this perspective, the user interface serves as a boundary between the algorithmic behavior of the tool and the goals and actions of the user: its primary function is in managing the information that enters and leaves the closed computational system. Although useful, perhaps essential to the practice of computer science, this view of user interfaces as mediating between a computational system and the world of human actions and intention also limits our ability to understand the factors that influence tool adoption.

For successful design, particularly of tools as complex as large-scale data visualization software, we must recognize that the interface is the user's primary experience of the system. From this perspective, the user interface defines a space of possible interactive behaviors within a larger context of human activities, goals, values, practices, and physical and social constraints. The interface remains a mediating artifact, but rather than mediating between user and algorithm, it mediates among the different members of the work community.

A compelling example of the complexity of this larger space of human activity comes from a recent discussion on the problems of analyzing large medical data sets that two of the authors attended. What was striking about this discussion is the way in which two MDs described their expectations for using a large database of medical information. After briefly going over a printed data definition sheet, the doctors shifted their attention from the structure and semantics of the data, to the complexities of using the data in a medical context, including the different uses of the data by doctors, nurses, hospital administrators, researchers, and IT specialists. It was evident in their discussion that this data was not just a source of objective medical information, but also played a role in decisions on managing the hospital, assigning responsibility for administering care, and meeting legal, economic, and regulatory obligations. As this illustrates, the function of the user interface is not simply to deliver algorithms and data; it must also mediate the complex flows of information, responsibility, negotiation, and activity that define communities of practice.

Even if a tool provides useful functionality and the user interface presents those capabilities effectively, if its interactive behaviors do not fit gracefully and intuitively into this larger space of

activities, then users will experience the tool as awkward, unreliable, and a poor fit to their goals and activities.

3.1 Philosophical roots

In moving our focus away from the cognitive and perceptual dimensions of usability to the process through which a community interprets, shares, and acts on data, interaction design has its roots in a philosophical tradition going back to C. S. Pierce [20], William James [21], John Dewey [22] and pragmatist epistemology, which acknowledges the complexities that lay between reality and our understanding of it. In particular, the pragmatists recognized that before information could be used, it must be interpreted, and that interpretation is shaped by the goals, culture, values, experience, and practices of the interpreter as well as the formal properties of the data. This does not deny the validity of empirical science; rather it emphasizes the difficulties in achieving an empirically valid, socially accepted understanding of the natural world.

The problems of interpretation are exacerbated by the difficulties users have in giving an explicit account of the way they use tools and information in their work practice. This reflects Heidegger's [23] phenomenological distinction between a tool designer's awareness of the details of a tool's form, materials, and function, and the user's experience of the tool in which the particulars of its design disappear in the experience of use. When a musician plays an instrument, they do not experience the instrument as a designed object; they experience music. Similarly when a doctor interacts with medical data, they do not experience algorithms and data structures; they experience patterns of symptoms, diseases, and treatments. Not only are people's ways of using tools complex and situation dependent, but also their ability to describe their own activities in terms that are specific enough to support software design is limited.

3.2 Implications for large data analysis and visualization

Early work in software design and usability was closely linked to the development of cognitive engineering, the idea of using the theories of cognitive science to improve the design of tools to support knowledge work [24]. In the 1990's, as personal computers became more powerful, applications more complex, and the Internet redefined computers as media, software designers became increasingly aware of the limits of the cognitive engineering approach, and shifted their focus from problems of perception, memory, and other cognitive constraints to the social, historical, and physical context of computer use. In a very real sense, the practical demands of designing software came face to face with the philosophical problems outlined above.

One of the earliest explorations of these ideas was Suchman's [25] study of usability problems with the first generation of "intelligent" copiers. This research demonstrated that people seldom follow a tool's intended use procedures, but continuously improvise their actions in response to their changing technical, physical, and social context. This insight is important to the problems of large-scale data analysis and visualization. Because scientific data analysis is an exploratory process, scientists require greater flexibility in their tools and procedures than traditional software design methods can address. These communities do not follow the kinds of explicit, controlled processes that support IT development traditionally assumes; rather, they continuously reconfigure their activities and organization in response to the contingencies of scientific exploration. This complexity has led designers to make greater use of ethnography and other field study methods [26] to achieve a deeper understanding of the specific community intended for system use.

However, field methods provide only part of the solution. The results of field studies must be interpreted to be of use, and this requires a strong theoretical framework. The common thread in theories such as distributed cognition [27] and activity theory [5] is that much of the structure of communities of work can be gleaned from an examination of the way tools and information *mediate* human activity. Rather than being neutral instruments, tools implicitly shape the ways we think of and perform work. This view supports a richer understanding of the dynamic, situational nature of exploratory knowledge creation, while providing a robust organizing structure for the design of tools to support it.

3.3 Activity Theory

Activity theory is a model of socially situated collaborative work based on Leont'ev and Vygotsky's [28] cultural-historical psychology, and the idea of cultural mediation: the way culture, as expressed through communication, artifacts, and behaviors, mediates the development and expression of our thoughts and actions. Leont'ev and Vygotsky's essential insight is that cultural artifacts such as tools, representations, and social norms both shape our patterns of thought and action, and are in turn shaped by those patterns. The problem we face in designing tools is not simply one of presenting technical capabilities. As designers, we also face the inevitability that the tools we build will change our user's ways of thinking and acting in ways we cannot anticipate – but that will potentially undermine the assumptions on which the tool is based.

Activity theory is not a predictive theory; it is a descriptive framework for analyzing socially situated behaviors that has proven useful in design. The central focus of activity theory is the *activity*: a system of actions taken to reach a goal in a particular context. The basic structure of an activity is that it is performed by an individual *subject* within a larger *community*, and acts on a specific *object* with the goal of producing an *outcome*. For example, individual analysts working on a research team perform the activity of finding patterns of interest in a large data set. The object of this activity is the data set, and the outcome is a deepened understanding of that data (figure 1).

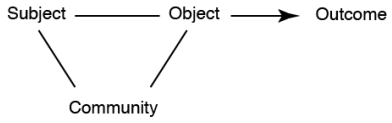

Figure 1

As described in section 3.2, an activity is mediated by tools, which include formal data sets and other information artifacts. It is also mediated by the roles established in the community, and accepted rules of social interaction (Figure 2). In the data analysis example, mediating tools would include visualization software, query engines, data management tools, and even general tools such as email. Roles would include analyst, database manager, algorithm developer, visualization designer, etc. Rules might include practices for sharing data and interpretations in the team, restrictions on who communicates with the customer, etc.

Although simple, when supported by systematic empirical investigation of a user community, activity theory can be a very powerful tool for understanding the interacting forces that determine how users will respond to a large data set and its associated tools.

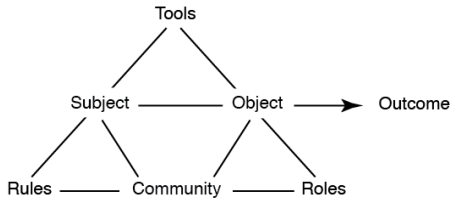

Figure 2

In the following section, we will use the concepts of activity theory to explain why a visual analytics software prototype was not widely adopted by the intended users, despite the design team's commitment to user-oriented design. Although the prototype was oriented toward smaller text corpora, the developers intended to scale the technology for significantly larger datasets. The adoption problems encountered at the prototype stage highlight issues for future iterations of this project. Activity theory provides a helpful framework for understanding why users did not perceive the software as useful, despite extensive input from members of the user community, and provides insight for incorporating context issues for data analysis and visualization tools.

4. The Scalable Text Prototype

Between 2008 and 2010, the authors were involved in a research and development project to develop a text analysis and visualization tool for text data. To preserve the confidentiality of the many people who supported this study, we will refer to this project as the "Scalable Text Analysis" (STA) effort, and use similar pseudonyms throughout this discussion. The project was intended to assess how scalable text modeling algorithms might be leveraged with visualization to eventually help analysts work more efficiently and effectively with large corpora of text documents.

The four authors were involved in various stages of the project, including early requirements and design, tool development, and an assessment performed at the end of the project, making it an unusually well studied development effort.

4.1 Background on the STA project

The STA project brought together members of two organizationally distinct communities – a computer science research group and a policy analysis group – who were located in the same large government institution, but whose work contexts did not intersect on a day-to-day basis. The policy analysis group was (and remains) well respected for its members' expertise in international science and technology policy. Similarly, the computer science research group was well-known for its expertise in scientific computing. In fact, members of each group had previously collaborated on policy analysis issues, and had even pursued some small-scale software development projects.

These interactions were fruitful enough that members of each community decided to pursue a more significant, higher-risk partnership to develop the Scalable Text Analysis project, which was oriented toward novel algorithms and frameworks for text analysis and visualization technologies. Members of each community saw complementary benefits: the research community would have an opportunity to pursue and evolve algorithmic research for text analytics, while incorporating these algorithms into novel information visualization environments. Similarly, members of the policy analysis community were intrigued by the possibility of text analysis tools that could help them manage increasingly large corpora of text information.

At the same time, project members in both communities were concerned that organizational differences between the two groups could present significant barriers to effective collaboration. The policy analysts were accustomed to producing written assessments of international trends in fairly short periods of time, and providing detailed assessments of emerging technology trends for a wide array of government consumers. Meeting project timelines, managing tight budgets, and producing usable policy reports for demanding clients were all extremely important for continued success of the policy group. In addition, the analysts were funded primarily on a project contract basis, with little flexible funding to support research or technology development projects. In contrast, the computer science researchers were accustomed to getting relatively large amounts of flexible research funding to pursue longer-term basic technology development activities. Its members were comfortable with a high degree of project risk, given the community's normative orientation toward innovation. In addition, the computer science researchers were used to working with publicly available information and publishing research results in the international scientific literature. In contrast, much of the policy analysts' data included sensitive government and proprietary information, and resulting reports were often sensitive or classified. Maintaining confidentiality around data, consumers, and topic areas was critically important for the policy analysts, just as publishing technology advances and research results was critically important for the research community.

The STA project was funded as a research project, but its goals included basic computer science research and development, as well as prototype analysis applications incorporating the project's algorithms. In a nutshell, the STA project researchers would develop new techniques and package these in prototype applications for deployment and testing in the policy community. Members of both communities realized, however, that significant cultural differences required careful management and negotiation if the project was to be successful. To ensure that the software would address the needs of the intended user community without violating any regulations around information management, leaders in the policy analysis and computer science groups decided to create an STA software prototyping team, which we will refer to as the "STP Team" or "Scalable Text Prototype" team. The STA project agreed to provide the policy group with direct funding to support two analysts as advisers to the STP project.

Over several months, the STP team – comprising policy analysts and computer scientists of various types, as well as an organizational anthropologist (McNamara, one of this paper's authors) – met on a regular basis and asked the analysts to describe their current approaches to working with text information, including data types, databases, current tools, and their policy research products. As the policy analysts described how they searched, reviewed, and incorporated electronic text information into their analytic work, the computer scientist researchers periodically asked the analysts to provide more detail on specific tasks and tools. The computer scientists then suggested ways in which novel text analysis and visualization approaches – many of which had been developed in a research environment but not deployed to external user communities – might support some of the text searching, retrieval, and categorization functions that the analysts described as routine elements of their workflow.

Over several iterations, the team members created a spreadsheet that conceptually linked analytic work tasks to research algorithms and used this as the basis for a more formal requirements document to support development of the STP software. The team's visualization researchers created several screenshot prototypes of a user interface for the STP software, and reviewed these paper prototypes with the policy analysts, who asked questions about functions that were unclear and suggested changes to the

interface layout. Once the team members agreed that they had reached consensus, the computer scientists began working on the STP software.

4.2 The STP Prototype

Several months later, the STP developers presented the policy analysts with the text prototype. The STP functionality included keyword queries within documents, querying across a document corpus, visualizing topical relationships among documents, and clustering documents into topical categories. To support these functions, the STP software incorporated several types of text analyses, including Latent Semantic Analysis, entity extraction, clustering, and contained graph or sub-graph searches. These algorithmic functions were coupled with several graphical representations of text content, including vertex-edge and tree ring graphs, to afford users more efficient access and insight into the content of the document cache.

Latent Semantic Analysis (LSA) is a natural language processing technique used to analyze relationships between a set of documents and the terms contained in the documents. Unlike techniques that parse sentences and map them into conceptual graphs or other representations, LSA represents the semantics of a document in terms of statistical properties of the words (or terms) in the document. This is accomplished by generating sets of topics related to the documents and the terms. LSA utilizes a term-document matrix, or frequency matrix, where rows correspond to terms found in the document cache and columns represent the documents. Each value in the matrix is proportional to the frequency of a term within the document (in an unweighted matrix each value would be equivalent to the frequency of a term within a document). Using these term-document matrices, STP uses GMeans clustering [29, 30] to form document clusters. A strength of these methods is their efficiency and amenability to parallelization, research goals of the project.

Because of the importance of proper names of people, organizations, and places to the analysts, STP also provides entity extraction using the Stanford Named Entity Recognition (SNER) tools [31]. This uses a statistical model that must be trained using manually annotated training datasets (the larger the training set, the more accurate the results). Training in a given domain, or document collection, does not always translate well into a new domain or with a new set of documents. This is true for both grammar based and statistically based entity extraction techniques; therefore, tuning the extraction algorithms is beneficial for all new data.

STP also utilized a variety of graph algorithms and heuristics that are useful for the analysis of network data. STP constructs graphs using document-to-document similarity measures (calculated with the LSA method described above), and document-to-entity relationships using the in-document relationships found by the entity extraction methods.

The policy analysts used the prototype on a trial basis. It quickly became apparent that they would not adopt the prototype. After informal discussions indicated that the analysts felt the prototype did not meet their needs, we were asked to investigate the cause of the adoption failure. This investigation followed two paths: a usability and visualization evaluation in our usability testing laboratory, and interviews with the analysts who tried the system and members of the development team.

4.3 Evaluations of STP

As part of this project, the authors of this paper were involved in evaluating the STP software. One set of evaluations followed the established pattern of usability studies. We conducted simple usability studies to assess the learnability of the software's functions. We also deployed prototype versions of the software to working policy analysts – both individual analysts and a project team – in the policy analysis community and observed them using the tool to review text datasets on their desktop computers.

Another series of studies emphasized issues particular to the analysis of large-scale data, and focused on the graphic representation of document networks. Conducted in a formal usability laboratory, these studies evaluated the suitability of different representations for different problems, as well as their scalability as the document corpus grew.

The final study was a field study, consisting of interviews with target users in which they were encouraged to talk freely and in depth about their experiences with the tool, the development process, and the development organization itself.

The final section of this report presents these results using an activity theoretic analysis to emphasize interactions among the different communities involved in development, and the way tools, social roles, and rules of behavior mediated both the users' experience of the tool and the development process itself.

5. An Activity Theoretic Analysis

There are a number of unique problems involved in designing and evaluating visualization software. The communities involved with the analysis of large-scale data tend to be large and multi-disciplinary, involving skills in computer hardware and software, data management, programming, and various application domains. In many cases, teams are geographically distributed. The application domains tend to be highly technical and specialized, but many particulars of data analysis are intertwined with domain specific knowledge, further complicating design. The rapid rate of change of technology in scientific fields is mirrored in a lack of stable or clearly defined work processes. This section examines these and related problems in the context of the STA project using activity theory as an analytic framework. Following the structure of figure 2, it divides the analysis into conclusions on the communities involved, the way the objects of work differed across subjects, problems with the way STP mediates work, and conflicts resulting from differences in rules and roles across communities.

5.1 Communities

Although leaders of STA characterized the effort as a single project, the work involved five distinct communities: The **computer science research** community was interested in developing informatics algorithms that could run efficiently on massively parallel computers. The **analysts** were looking for tools that could help them address the problems of analyzing large corpora of documents in short time frames. The **developer team** was interested in developing and testing informatics libraries that could be used to build a variety of applications. The **usability group** worked closely with the user community to develop potentially useful prototypes and to evaluate their success. Finally, the **funding office** managed internal R&D funds, and was charged with funding exploratory research with the potential for high long-term benefits.

It is interesting to note that this is a common organization for advanced software development projects. The division between research and development reflects the common model of an innovation pipeline, and the development team recognized the value of usability and the need for specialists in this area. What this functional organization, however intuitive, overlooked in this situation was the differences in goals and culture across these subteams.

For example, the development and usability teams focused on delivering a series of prototypes to the analysts. Although the analysts recognized that these would be research prototypes with

limited support and less robustness than production software, everyone agreed on the importance of releasing prototypes regularly, and refining them across revisions in keeping with analyst feedback. In contrast to this commitment to a regular schedule of deliverables, the research team was focused on developing mathematical algorithms and publishing on their results. This requires a more dynamic response to an open-ended series of research problems, in which the solution of one problem opens new research opportunities. Although this work was punctuated by deadlines for conference papers, the overall flow of work was more dictated by research opportunities than project milestones.

This difference between the research and development communities exerted a subtle influence on the prototyping activities of the development team. Rather than using successive prototypes to refine the delivery of a single set of capabilities, they came under pressures to introduce new research ideas into successive prototypes, which further prevented the development team from focusing on turning a single capability into a useful, usable system.

This also led to conflicts over funding, with the project leadership at one point appropriating funds from the development team's budget to pay for new research opportunities. This problem was exacerbated by incentives in the funding office, which faced immediate pressures to maintain the quality of research and publication, which took precedence over the equally important but less immediate goal of long-term applicability.

These kinds of conflict are certainly familiar to anyone who has worked on software teams, and we recognize the difficulty in anticipating them in a project, particularly when the result from widely accepted practices. However, we do hope this analysis calls attention to the subtleties of these issues, the way they the progress of technical work, and the extent to which domains like large-scale data analysis and visualization exacerbate these problems. We also hope that this discussion will replace the common view that human problems are inevitable, with a recognition that the organization of project teams is itself an object of design that has a mediating effect of collaboration, and that the design of organizations can also be informed by analyses of this type.

5.2 Object conflicts

Most of the conflicts outlined in section 5.1 could be thought of as differences in the objectives of the differing work communities. Too often, project leaders lament culture conflicts across subteams and the difficulties of addressing something as implicit and diffuse as culture, but arguably, "pure" culture conflicts may be less important than conflicts in the objects of work done by different groups. This idea of aligning objectives across subteams, as well as between the developers and user community is critical to effective design, and can be surprisingly subtle. A particularly interesting example of this is what we believe was a misunderstanding of the basic object of the analysts' activites by the software developers.

One result of the field study of STP's users was a consensus among analysts that the tool was difficult to understand, and the document categorizations produced by LSA did not make sense. These observations included a feeling that the tool did not present a clear conceptual model of its functionality, and failed to incorporate an understanding of the analyst community into its design.

We believe this reflects a subtle difference in the approach to problems taken by the analyst and computer science communities. Members of the STA team were focused on solving problems, which, although complex and as yet unsolved, did have a clear technical statement, and relatively clear criteria for an acceptable solution. In contrast, the analysts' work is better explained as a form of *sensemaking*: a process of finding order in data that may not have an inherent order, and doing so within limited time and resources, multiple social constraints, complex ethical implications, and some degree of personal and organizational risk. Karl Weick [32] defines sensemaking through comparison to a codebreaking game called Mastermind:

> *The object of Mastermind is for a codebreaker to duplicate the exact pattern of colored pegs inserted into holes that has been created by a codemaker but is concealed from the codebreaker by a shield. The codebreaker ventures hypotheses as to what the pattern might be and, on the basis of information supplied by the codemaker, refines the hypothesis until the codebreaker's hypothesis exactly matches the codemaker's original pattern.*
> *Mastermind is precisely what sensemaking is not. People cannot be sure there is a mastercode to be discovered, nor can they be sure what the nature of the code might be, nor even if there is some order in the first place. Even if people do discover the code, they can never be sure they have done so since there is nothing equivalent to the removal of the shield at the end of the game, which reveals the concealed code. [32] page 8.*

As this suggests, and as conversations with analysts confirm, the analysis process has strong elements of sensemaking, where document collections may be incomplete and unreliable, customer questions may be ambiguously framed, uncertainty is inherent in all stages of analysis, no clear "right" answer may be possible, and the consequences of error can be severe. Sensemaking depends critically on a broad range of resources including the analyst's training, personal experience, social context, character, goals, customer relationships, human capacity for empathy, and even the ethical issues associated with recommending critical actions on the basis of imperfect information.

When considered from the perspective of how tools can best mediate the sensemaking process, limitations of the purely syntactic methods such as LSA led to a sense among users that the tool was not relevant to their work. This is not to suggest that syntactic search methods are of no use in interpreting large-scale data sets; indeed, the size and complexity of this data often means that these are the only options available. However, it does underscore the importance of designing tools to present the algorithm's functionality not only accurately, but also in ways that fit the user's understanding of their problem domain, and that insure users have an adequate sense of the risks and limitations of the tool. The next section on tool mediation addresses these problems.

5.3 Tool mediations

A central idea of activity theory is that tools mediate the larger fabric of activities in a user community. One finding of the user field study that emphasizes this role was users' feeling that the STP prototype did not adequately support the analysis lifecycle. The prototype was a stand-alone tool that delivered its algorithms, but did not support interoperability with other tools, such as statistical packages, or allow users to copy results into presentations or reports. In addition, it provided weak capabilities for saving results, which limit its ability to build on past analyses to improve performance over time. Although the development community felt that these more mundane functions were less important than the ability of the algorithms to create an order in the documents, from the analyst perspective, they reflect a critical inability to mediate the overall flow of their work.

A related complaint was that the tool offers limited forms of interactivity. It constructs networks of related documents, individuals, locations, and organizations, but does not allow users to modify these networks interactively. This makes it difficult for analysts to bring their own knowledge to bear in guiding document cluster-

ing or entity extraction, and leaves the tool unable to support the iterative process of proposing and refining alternative interpretations that is central to sensemaking forms of analysis.

The importance of the mediation perspective is also evident in laboratory evaluations of the user interface. Figure 3 shows the STP interface. Although the details are difficult to distinguish in this small figure, several points are significant. The pane in the upper left hand corner lists the clusters the tool created in a hierarchical format; clicking on a cluster brings down a list of the documents in that cluster. Selecting a document from this list causes it to be displayed in the large center pane. The lower left hand corner displays the documents in a ring diagram (see Figure 4), with documents arranged around the perimeter of the ring, and connected by arcs across the center. Documents in the same cluster were given the same color.

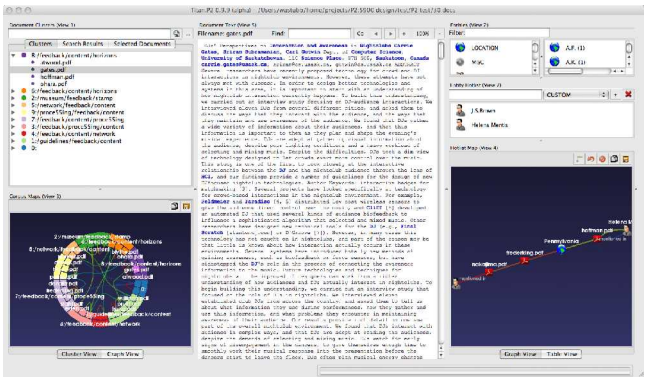

Figure 3

This is a logical layout of the tools' functionality, but closer evaluation indicates weaknesses when evaluated in terms of its ability to mediate collaborative sensemaking. The layout is logical in arrangement, but only when considered as a representation of the underlying algorithms and data structures. Each pane represents a logical component of the tool's organization, rather than the structure of sensemaking in the problem domain. For example, the center pane allows display of a single document, but only allows viewing of one document at a time. This is a poor fit for the way people make sense of document sets, which involves comparing documents. Also, the display reflects the algorithm's process of stripping text out of pdfs and other formatted documents for analysis. From an analyst's perspective, however, things like bolded fonts can provide valuable clues to an author's intended meaning. Also, opening the same documents outside the tool will result in a very difficult appearance, which can confuse the analysis process.

As theories of mediation suggest, visualization tools are not neutral translation of data into images, but carry analytic assumptions of their own. One of our evaluations of STP as a mediating artifact compares the ring diagram of document networks (Figure 4) with a representation of them as networks of nodes and arcs (as in the bottom, right-hand pane of figure 3). This was done in a controlled laboratory environment, where subtle differences in short term memory demands, error rates, and other factors could be measured. The result showed a marked difference in effectiveness, and scaling behavior of each representation on tasks such as finding related or unrelated clusters, or finding documents that connected to documents in many other clusters. Although space prevents going into this work in detail, it underscores the importance of controlled evaluations of visualizations to understand the biases they introduce into the interpretive processes they mediate.

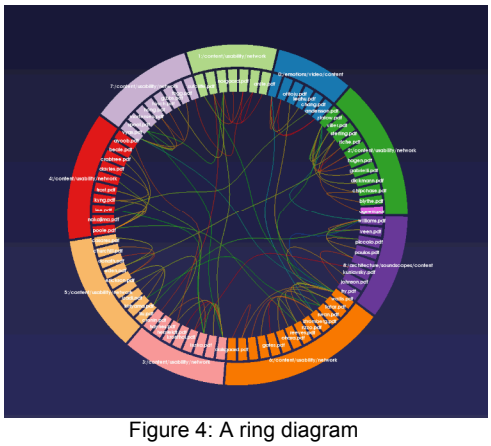

Figure 4: A ring diagram

5.4 Rule mediations

Rules of social behavior provide another form of mediation that can affect adoption of a tool. One source of difference between the analyst and developer communities was in restrictions on the ways they handled funding and customer interactions. As a research project, the leaders of the STP team were given considerable autonomy in how funds could be spent; in contrast, the analyst community worked on a fee for service basis where customer funds were allocated for specified tasks, and spending was closely tracked. The development team did not initially place adequate emphasis on the role of funding in shaping priorities, and as a result, were often seen as unresponsive. As the project neared conclusion, and discussions were held on future work, the analysts emphasized importance of controlling funding to insuring their needs were met. Similarly, the analyst team often worked with customer proprietary information, whereas the research team was used to a more open culture of information sharing. The short time frames commonly allowed for of analyst work also contrasted with the longer time frames for research. Although these differences in social practices did not cause major problems in the groups' collaborations, they were the source of numerous adjustments among participants. In addition, they raise questions about the affect of these rules on the analyst's sensemaking processes. Although tracing these effects would additional investigation, this initial characterization further illustrates the subtlety of social rules and their effect on adoption of tools.

5.5 Role mediations

In our evaluations, we also noted a confusion of roles in the project. As discussed in 5.1, although the STA project followed a standard functional organization into subteams, this led to conflicts among subteam goals and objects that affected work with the analyst community and hindered adoption. It is also worth noting that no one on the team had both the responsibility and resources to speak for the customer. The usability team did have access to the users, and represented their interest, but their role was peripheral to the primary decision making and funding authority in the project.

6. Conclusion

It is often said that there is much to be gained from discussions of efforts that did not unfold as hoped for, but few forums are willing to discuss such results, and researchers are even less eager to publish them. The research discussed in this paper was, we are happy to say, anything but a failure in the larger scheme of things. Although the specific prototype discussed did not secure adoption,

the people involved are continuing to work together to develop informatics analysis and visualization tools to support the analysts' work. In addition, both management and developers have been receptive to the observations made in this paper.

We hope readers will take from this analysis a deeper appreciation of the subtle social and contextual factors that influence adoption of new technologies that are so critical for large-scale data analysis and visualization, and also will see the value in activity theory and other models of the way in which tools, information, and social structures mediate collaboration and influence the adoption of advanced technology.

As a final comment, we note that the organization and processes of a research and development team are, like the tools they construct, objects of design. Tools like activity theory are just as applicable to the design of organizations as to the design of tools, and afford an approach to solving some of the problems outlined in this paper.

ACKNOWLEDGEMENTS

The authors wish to thank Tim Trucano for his invaluable counsel in developing the ideas presented in this paper, Sandia Labs for supporting this research, and everyone involved in the studies discussed herein for their willingness to participate in the kind of honest reflection on work that is so critical to improvement and the advance of knowledge.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94aAL85000.

REFERENCES

[1] Hey, T., Tansley, S. and Cole, K. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, City, 2009.

[2] Ma, L. K. Large Scale Data Visualization. *IEEE Computer Graphics and Applications*, 21, 4. 2001. 22-23.

[3] Fox, P. and Hendler, J. Changing the Equation on Scientific Data Visualization. *Science*, 331, 6018. 2011. 705-708.

[4] Samatova, N. F., Breimyer, P., Hendrix, W., Schmidt, M. C. and Rhyne, T.-M. *An Outlook into Ultra-Scale Visualization of Large-Scale Biological Data*. IEEE. 2008.

[5] Nardi, B. A. *Context and Consciousness: Activity Theory and Human-Computer Interaction*. MIT Press, Cambridge, MA. 1996.

[6] Card, S. K., Mackinlay, J. D. and Shneiderman, B. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann. 1999.

[7] *Map of the Market*. Wall Street Journal Digital Network, 2010.

[8] Johnson, B. and Shneiderman, B. Tree Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Strustures. In *Proceedings of the VIS '91 Proceedings of the 2nd Conference on Visualization '91*. 1991. IEEE Computer Society Press.

[9] IBM Cognos Software Group *Many Eyes*. IBM Research. 2011.

[10] Shneiderman, B., Rao, R., Andrews, K., Ahlberg, C., Brodbeck, D., Jewitt, T. and Mackinlay, J. Turning Information Visualization Innovations into Commercial Products: Lessons to Guide the Next Success. *IEEE Symposium on Information Visualization*. 2005.

[11] Craft, B. and Cairns, P. Beyond Guidelines: What Can We Learn from the Visual Information Seeking Mantra? In *Proceedings of the Ninth International Conference on Information Visualization*. 2005.

[12] Shneiderman, B. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualization. In *Proceedings of the IEEE Symposium on Visual Languages*. Boulder, CO. 1996.

[13] Bertini, E., Plaisant, C., Perer, A. and Santucci, G. BELIV'08: Beyond Time and Error: Novel Evaluation Methods for Information Visualization. In *Proceedings of CHI'08*. Florence, Italy. 2008.

[14] Bertini, E., Plaisant, C. and Santucci, G. BELIV'06: Beyond Time and Errors: Novel Evaluation Methods for Information Visualization. *Interactions*, 14, 3. 2006. 59-60.

[15] Munzer, T. A Nested Model for Visualization Design and Evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 15, 6. 2009. 921-928.

[16] Shneiderman, B. and Plaisant, C. Strategies for Evaluating Information Visualization Tools: Multi-dimensional In-depth Long-term Case Studies In *Proceedings of the Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization*. Venice, Italy. 2006. ACM Press.

[17] Bertelsen, O. W. and Bodker, S. *Activity Theory*. Morgan Kaufman, City, 2003.

[18] Logren, J. and Stolterman, E. *Thoughtful Interaction Design: A Design Perspective on Information Technology*. MIT Press, Boston, 2004.

[19] Preece, J., Rogers, Y. and Sharp, H. *Interaction Design*. John Wiley and Sons, New York, 2007.

[20] Weiner, P. P. *Charles S. Peirce: Selected Writings*. Dover, City, 1966.

[21] James, W. *William James: Writings 1902-1910*. Library of America, 1988.

[22] Hickman, L. A. and Alexander, T. M. *The Essential Dewey*. Indiana University Press, 1998.

[23] Heidegger, M. *Being and Time*. Harper & Row, New York, 1962.

[24] Carroll, J. M. *Encyclopedia chapter on Human Computer Interaction (HCI)*. Interaction-Design.org, City, 2009.

[25] Suchman, L. *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press, Cambridge, 1987.

[26] Laurel, B. *Design Research: Methods and Perspectives*. MIT Press, 2003.

[27] Hutchins, E. *Cognition in the Wild*. MIT Press, Cambridge, MA, 1995.

[28] Vygotsky, L. S. *Mind in Society*. Harvard University Press, Cambridge, MA, 1978.

[29] Dhillon, I. S., Fan, J. and Guan, Y. *Efficient clustering of very large document collections*. Kluwer Academic Publishers, 2001.

[30] Dhillon, I. S., Guan, Y. and Kogan, J. *Iterative clustering of high dimensional text data augmented by local search*. City, 2002.

[31] Group, S. N. L. P. *Named Entity Recognition (NER) and Information Extraction (IE)*. 2005.

[32] Weick, K. *Making Sense of the Organization*. Blackwell Publishing, 2000.