The Social Life of Engineering Authorizations

William A. Stubblefield
Sandia National Laboratories
Albuquerque, New Mexico 87185 USA
1-505-284-2856
wastubb@sandia.gov

Karen S. Rogers
Sandia National Laboratories
Albuquerque, New Mexico 87185 USA
1-505-845-0837
ksroger@sandia.gov

ABSTRACT
We may view documents, not only as "containers" for information, but also as active participants in organizing and sustaining communities. This paper discusses our experiences in designing a web-based tool for writing and managing engineering authorizations, and the social perspective's influence on our understanding of the problem and the design of our system. It presents observations based on our fieldwork with users, and the evaluation of a set of prototype systems. It shows how these observations changed our central metaphor for the system, moving it from a machine model to a "society of agents" metaphor. Finally, it illustrates the way this new metaphor changed our system functionality and architecture

Keywords
Documents, community, design ethnography, metaphor, system design.

1. INTRODUCTION
> *For documents are much more than just a powerful means of structuring and navigating information space – important though that is. They are also a powerful resource for constructing and negotiating social space.*
>
> *The Social Life of Documents*
>
> J. S. Brown & P. Duguid

Although we generally think of documents as containers for information, we may also characterize them in terms of their social functions. In *The Social Life of Documents* [1], Brown and Duguid examine how documents help define and sustain a community. Documents identify a community's members, articulate shared values and beliefs, mark its borders, frame debates, authorize and motivate individuals to act, and structure the negotiation of their own meaning.

LEAVE THIS TEXT BOX IN PLACE
AND BLANK

This paper recounts the social point of view's influence on our design of a web-based tool for creating, distributing and storing engineering authorizations. It discusses the unexpected behaviors of engineering authorizations it revealed, behaviors that otherwise might have gone unnoticed. It discusses its influence on our design, and the problems and opportunities it brought to our attention, many with which we are still contending.

2. A NOTE ON METHODOLOGY
As working software engineers, our methods are a curious, but not atypical, mix of theoretical ideas and practical necessity. The observations in this paper are very much the product of such a mixture. We characterize our methods along two dimensions:

- The first dimension concerns when prototype systems are constructed. This varies from approaches, like the classic waterfall method, that delay construction until after an extensive design phase, to rapid prototyping techniques that start implementation early. We began building software very early, using a series of prototype systems as a focus for many of our user and customer interactions. Prototypes ranged in sophistication from medium fidelity screen prototypes (essentially Power-point mock-ups of proposed screens) to almost fully functional web clients.
- The second dimension concerns the role of users in the design process. At one end are approaches that view users as subjects of systematic study by the design team, using structured observation and interview techniques. Design ethnography [5] and techniques like Distributed Cognition [6] tend toward this end of the spectrum. At the other end are participatory design methods [7] which strive to eliminate the subject/object distinction between users and designers, and involve users directly in the design process. Our own work fell in the middle of this continuum. We began with observations and interviews, but we began building prototypes very quickly. Most of our interactions after the first month or so involved users more directly in the design process, using the prototypes as a common focus.

We think of this approach as *design driven*, for its reliance on the design artifact as a focus for interactions with the user community. We chose this approach for pragmatic reasons. Although we strongly believe that all design work should build on an empirical understanding of our users' needs, assumptions and constraints, in practice, organizational and financial limits kept us from doing as much pure fieldwork as we would have wished. All the users we contacted were enthusiastic and will-

ing to help us, but they were limited in the time they could allocate to our needs. Similarly, to secure and protect our funding, we needed to show results quickly. These factors pushed us toward a mix of fast prototyping and user involvement.

This approach did have some unexpected benefits. One of the most important was the way prototypes helped us tie together the views of an extremely diverse set of users, customers and stakeholders. As the rest of this document will make clear, our project effects the lives of engineers, document managers, business process designers, security people, and management at both the labs and the Department of Energy. I doubt that we could have addressed the often-conflicting viewpoints of these communities without the common frame of reference afforded by our prototypes. The remainder of this paper discusses what we have learned from our design driven approach to understanding the function of engineering authorizations within and across these diverse communities.

3. BACKGROUND

For the last two years, we have designed systems for managing engineering tools and information. The main focus of our efforts has been on *document agents*: active computational objects that not only contain information, but also participate actively in its creation, maintenance and transmission. We define a *document* as a body of information with a clearly defined frame or boundary. Under this definition, a document can range in complexity from simple files to complex assemblies of solid models and supporting information such as component libraries, analysis tools and test data. A *document agent* combines a document with the algorithms used for creating, editing, storing, validating and communicating it across its life cycle. Document agents do more than apply object-oriented programming to document management: we have also used agency as a user-interface metaphor, enabling us to present complex document structures and functionality to users in an intuitive manner. We first developed this approach in a prototype for a Printed Wiring Board Design Environment, using document agents to organize the complex capabilities required for designing and fabricating wiring boards into an integrated system. In October 1998, Sandia Laboratories asked us to apply our methods to the management of engineering authorizations.

An Engineering Authorization (EA) formally authorizes an organization to take some action in the design or manufacture of a mechanical part or assembly. EAs both authorize specific actions, and also document the history of actions taken, problems solved, and decisions made. The National Laboratory System uses EAs to coordinate the activities of design and production facilities scattered throughout the United States. The requirements for their content and handling originate with the US Department of Energy (DOE), which manages the national laboratory system. Although, the labs use over a dozen different types of EAs, these fall into two main groups:

- **Engineering Releases (ERs)** release a specific version of a design to a production facility for manufacture. Actions an ER authorizes include tooling a plant for production, building evaluation prototypes and full-scale manufacture. An ER also designates a set of drawings or other documents that constitute the artifact's definition of record.
- **Change Orders (COs)** authorize modifications to an artifact's definition of record. They specify such things as the nature of the change, the reason for the change and the date it takes effect. Final Change Orders (FCOs) document changes that have already been made to an artifact's formal definition: essentially, they notify people that a modified design has been released. Advanced Change Orders (ACOs) describe changes that have not yet been made to the definition of record, and are useful where the production facility requires immediate authorization to change a product specification. It is understood that the changes specified in an ACO will eventually be incorporated into a new design release.

The history of Engineering Authorizations at Sandia National Laboratories begins with a manual, paper-based system developed in the 1950s and 60s. This system defined the basic types, content and textual format of EAs, along with the steps required for their approval and distribution. It required a large staff to perform these functions. Validation checked both syntactic (e.g. are all required fields filled in) and semantic properties of EAs (e.g. do the drawing numbers listed actually refer to the appropriate engineering drawings), and was done manually. In the early 1980s, the laboratories implemented a system that distributed and stored authorizations in electronic form. Content and format remained the same as in the old system: an EA was a highly structured text document. This system used a separate relational database of document meta-data as an index to the electronically stored EAs. This database allowed document managers to search for authorizations on such items as the author, type, date, etc.

Although an improvement over the old, entirely manual system, this second-generation system still required significant manual and paper-based support. It transmitted documents to recipients across the national laboratory system through a combination of electronic and paper-based transmission. The hybrid nature of distribution was a consequence of the wide variety of organizations subscribing to the system, differences in technologies, the complexity of business rules, and the strong security requirements mandated for much of our data. This second-generation system required a relatively large human document management staff to handle validation, tracking, storage and retrieval, although it was much smaller than in the old system. There were three reasons for this: 1) the database interface was complex and cumbersome to use, requiring special training; 2) the database required a computer loaded with special software, and could not be easily accessed over a network; and 3) due to their complexity, engineering authorizations still required validation by human experts. Owing to the conservative nature of our customers (the Department of Energy and the military), this system remained in place until recently.

In 1998, the labs began work on a new, fully automated system. This system, called EBOM (Engineering Bill of Materials) would store Authorizations, Engineering Drawings, materials lists and other part defining documents using an object-oriented document management tool manufactured by Matrix One, Inc. The Matrix One tool integrates document management (version control, format control, and life-cycle management) with an object-oriented database (to store document properties, search-

able meta-data, and software triggers). We were brought into the project to develop a tool that would allow engineers to create, store and track EAs from their desktop. Project management also hoped that our system, named the *EA Manager*, would provide syntactic and semantic checks on EAs. They hoped it would shift some of the burden of handling EAs from the document management staff to the engineers, and reduce the time it took to distribute an authorization from days to hours or minutes.

We began our design effort by looking at the laboratories' official documentation of the business practices surrounding EAs [2]. Although this defined the form and content of EAs, it did not address their social function. Given our belief in software's social and political dimensions (see [3]), we felt this viewpoint was essential, and pursued it by interviewing and observing a variety of potential users. Our main focus was on the engineers who create EAs, and who will be the main users of our tool. We also interviewed document management people who use the current system, authors of Sandia's technical business practices, engineering quality control people and designers of the new EBOM database. Our methods included interviews with single users, observations of engineers creating EAs under the current system, and focus groups of engineers and software designers. Often, these interactions focused on prototype systems we developed. Throughout this effort, we made a strong commitment to understanding engineering authorizations as both informational and social artifacts.

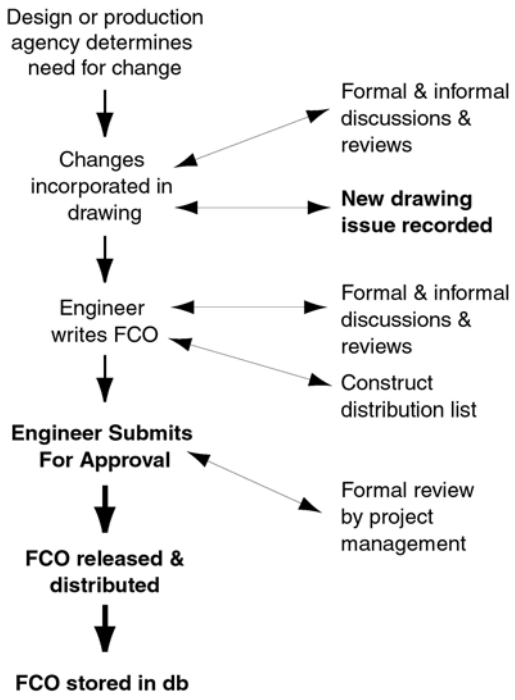
4. SOCIAL LIFE OF ENGINEERING AUTHORIZATIONS

Engineering authorizations are an ideal subject for the social analysis of documents, given that their primary purpose is to authorize actions in an engineering community. EAs also serve as historical records of the reasons actions were taken and the people responsible for them. For example, an engineer might change the tolerances on a part to improve manufacturing yields. The reason for the change (improving yields) is not noted on the drawing, but in the associated change order. This historical function supports future engineering efforts by recording design rationale, and provides tracing and accountability in the event an artifact fails. Given the high costs of failure for components used in military applications, this historical function is extremely important to our customers.

Another important feature of engineering authorizations is the complexity of their life cycles. As recorded in the laboratory's official business practices, the life cycle of an EA appears straightforward: an engineer writes it, after getting appropriate approvals, he sends it to the document managers, who send it to the distribution list and store it in the document database. As we discovered in our fieldwork, the actual life cycle of an EA was much more complex, and relied upon a combination of formal and informal systems, personal relationships, and networks of responsibility that were neither documented in our business practices, nor supported by the document management system. Finding a way to extend the scope of the EA Manager to automate more of the document life cycle, without damaging what we believe to be a vital and fragile web of social relationships and personal responsibilities, became one of our primary goals. In forging our understanding of these issues, the social perspective on documents proved invaluable. We discuss some of the specific issues it revealed to us in the following sections.

4.1 Integration of EAs and Social Processes

Figure 1 shows the life cycle of a Final Change Order. Although fairly high-level, the figure shows both the actual FCO life cycle, and those steps supported by the current semi-automated system (indicated in bold font). Many of the other steps are supported by other automated and human systems. For example, engineers and management draw on both automated systems and formal business practices when reviewing changes to a drawing. Also, the document management staff have designed and distributed Microsoft Word™ templates to assist engineers in writing EAs. Still other tasks have found no adequate support from automated or human systems, and remain problem areas: one example of this is the maintenance of distribution lists, which are either created manually, or copied out of old EAs and pasted into new documents. This is an important area we can automate, and quickly became an important focus of our design. Our approach draws on existing corporate personnel databases and mailing lists to better support this function.



Final Change Order Life Cycle

Figure 1.

Although not obvious from Figure 1, there is an interesting side effect of the current system. After an Engineer has received approval for a change order, the document management staff submits it to a secure distribution network. This system sends it out to its distribution list, officially releases it, and stores it in the database. Unfortunately, the system designers made the act of distribution and the act of releasing the same: once distributed, the EA is released. This makes it impossible for engineers

to distribute a draft version for comments using the official system. Instead, they handle informal reviews in an ad hoc manner: through e-mail, sharing files on a network and paper copies. In this sense, the system for managing EAs did a poor job of supporting the community that creates and uses them. As a result, we suggested that the new system allow documents to have a draft status, allowing draft documents to be distributed without officially releasing them, letting engineers use the system for informal reviews, co-authoring, etc. We hope this will do a better job of integrating the system into the community.The life cycle of an Advanced Change Order is much more complex (Figure 2), and illustrates the flexibility with which a community will adapt to fill in the gaps in a formal system. As in Figure 1, the actions of the formal system are in bold font; note that they are the same for ACOs as for FCOs. The source of an ACO's added complexity is that it specifies changes to a part definition before those changes have been recorded in the associated drawings. An ACO's main utility is in providing rapid response to problems by authorizing steps to be taken without first requiring those changes be made to the part definition. Consequently, ACOs are very popular with engineers. After an ACO has been distributed, the part's definition of record becomes the old definition plus any outstanding ACOs. This situation not only complicates drawing management and introduces the potential for errors, but also it persists until engineers or draftsmen have incorporated the changes specified in the ACO into the design, and officially issued a new drawing release. All users recognize this problem and support its repair.

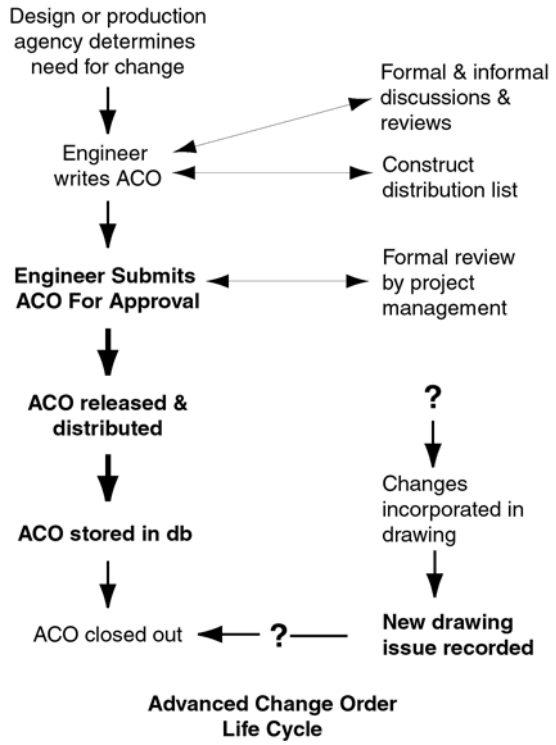


Figure 2.

Figure 2 illustrates additional problems with ACOs: outstanding ACOs are resolved when designers incorporate the changes into the part's definition of record. However, as the "?" in the figure indicates, there is no formal notification capability in the current system to make sure that this gets done. Also, once the changes have been incorporated to a drawing, there is no formal notification system to indicate the ACO has been closed. Document management staff have developed a solution to these problems independently of the formal EA management system, using small, individual, personal computer databases to keep a record of drawings with pending ACOs. Responsibility for incorporating the changes into the drawing remain with the engineer who authored the ACO, with the document manager functioning as a second line of defense and contacting engineers should too much time pass before this is done. Although this illustrates the community's robustness and ability to compensate for deficiencies in a formal system, the approach is error prone. Everyone agreed the new system should solve this problem.

Further investigation revealed the difficulties in formalizing this process. Human document managers know the engineers, their constraints and priorities. They understand when an engineer has good reason to let an ACO remain unincorporated into a drawing; e.g. they might wait for planned future changes to be defined. Also, document managers know which engineers tend to overlook such responsibilities. Consequently, they carry out the task of reminding the engineers with more discretion than an automated system. Neither the problem of distribution list management, nor the problems inherent in the life cycle of ACOs were documented in either the laboratory's documented business practices, nor could they be inferred from an inspection of existing ACOs. Discovery of these problems was a direct result of our commitment to looking at EAs as actors in a social context.

4.2 The Structure of the EA Community

Because of the importance of engineering authorizations, a complex structure of personal and organizational relationships has grown up around the EA system. For example, although the engineer who authors an EA is responsible for its distribution, the only people who have hands-on access to the current distribution system are the document management specialists. Many of the to problems with the current system, such as the resolution of pending ACOs, require direct participation of document management people. Consequently, most of the engineers we interviewed have developed strong, positive working relationships with document managers, viewing them as partners, rather than just data management specialists. This is significant. As one engineer put it: "I don't like the idea of an automated workflow manager, it makes it sterile, it takes the human factor out of a project."

The system's heavy reliance on personal relationships was one of the reasons management wanted a web-based system that would let engineers track EAs directly. Managing EAs is a difficult skill, requiring an understanding of complex practices and regulations. The retirement of a qualified person can create problems for the whole system. Also, because there are only a few qualified document managers, management perceived their participation to be a bottleneck for system efficiency. However, in the course of our fieldwork, we noticed other functions played by data managers. As one data management specialist indicated to us, her job really has two aspects: One of her responsibilities is to work as a "spell checker," performing syntactic and semantic validation on EAs. Her other responsibilities include problem solving or "detective work," helping engineers perform unusually difficult queries, helping them find people at other laboratories who can help them with problems, and generally greasing

the bureaucracy to get things done. She also indicated that training people is an important part of her job: she does not just solve problems for engineers, but also tries to teach them to use the system better. She agreed with the importance of automating her "spell checking" function, but correctly pointed out that we could not automate all her problem solving functions. The engineers largely agreed with this appraisal and characterized the document management staff as partners in the EA process who not only solve problems, but also help them avoid pitfalls with such complex issues as document security, changing government and corporate requirements, and changing staff and organizational structures.

This is doubly important in light of the fact that, owing to the devastating consequences of a potential failure of military, aerospace or national infrastructure technology. Sandia has always exhibited a strong culture of personal responsibility. When confronted with a difficult issue with a part or design, people's first question is invariably "who is the responsible engineer?" Taking personal responsibility for difficult tasks is so deeply ingrained in Sandia's engineering culture that it continues to resist efforts to institute more formal, process-oriented project management styles. As a result, rather than just administering a database, the document management people function as gatekeepers to a body of knowledge that is complex, exacting and critical for our national security, and infrastructure. Their personal authority is a valuable source of trust in the system. Although everyone involved, including the data managers themselves, felt that the current system had too many bottlenecks and needed to be streamlined, we are reluctant to support fully automated solutions that would exclude these people from the process. Even if our system could assure the correctness of EAs, we doubt that a fully automated system could win user's trust in a community so deeply committed to an ethic of personal responsibility.

Our investigation also revealed a complex and interesting structure to the communities involved in engineering authorizations. As figure 3 illustrates, at least three communities have an interest in engineering authorizations: engineers, document managers and the Department of Energy. The figure illustrates the relationships between them, with the document managers mediating the relationship between the engineers and DOE requirements. The figure also lists a number of differences in the goals, culture and composition of these communities that account for much of the complexity of handling engineering authorizations. Engineering is organized along team lines: teams are small groups of highly interrelated people, with functional responsibilities defining team membership, e.g. if needed, a project might form a metallurgy team. Teams assemble into larger teams, but these retain the flexibility and functional organization. In contrast, DOE is a classic bureaucracy, its structure is essentially permanent, and restricts interactions among members. Document management consists of small teams or individuals working to mediate the relationship between engineering and DOE. This small team size fits well with Sandia's culture of personal responsibility, giving engineers a responsible person, rather than a bureaucracy, with which to interact.

Engineering's primary goal is solving customer problems through technical innovation; in contrast, administration (DOE) is concerned with tracing the design and production of artifacts to guarantee compliance with government requirements and ultimate accountability. Engineering's values are pragmatic: their goal is to get the job done by the best means possible, and they will improvise if the situation requires it. Administration's value system is conservative: they require stable, controlled methodologies to insure the accuracy and longevity of documentation. It is interesting to note the properties of document managers that enable them to mediate these often-conflicting goals and values: although they share administration's goals, their working values are very pragmatic. Document managers will "do what it takes" to help engineers get their job done, routinely doing things like helping engineers write EAs, helping them find people for a distribution list, or using their personal knowledge and contacts to smooth out problems with the system.

Another interesting aspect of this community structure is that the primary people responsible for EAs at both sites we have visited are women, while most of the engineers and people specifying procedures and regulations are men. This is not an accident: the document management role grew out of the clerical functions of the original, paper-based system, and clerical work has traditionally been a female role. However, although we can only speculate, it is possible that their gender, by making them less threatening and fitting traditional female roles, makes it easier for document managers to assert necessary rules to the engineers with minimal conflict, and to act as mediators between these predominantly male groups. Although we encountered no overtly political issues, this introduces a feminist perspective to our work that demands our sensitivity. For example, we were concerned that the women who actually handled document management were not included in the early planning stages of the system, and used our influence to bring them into the process.

As this suggests, document managers play an important and subtle role in the engineering community. Because of the risks involved in altering the basic networks of trust, responsibility and knowledge in any community, we are reluctant to change this too abruptly. Consequently, we favor a solution that does not attempt to eliminate the document management people from the process, but rather one that reduces their role as "spell checkers" while preserving their function as guarantors of data integrity and mediators between these communities. Although this issue is still being debated and remains unresolved at this time, our emphasis on prototyping and user-oriented design gives the engineering community the opportunity to shape the system and related practices. If we are correct, we believe users will require continued involvement of human data managers in the new system.

4.3 Barriers to New Engineers and Outside Community

Another of a document's social functions is to define community membership, both by giving members a shared body of knowledge, and by excluding people who do not have access to the documents. Under the current system, EAs function in both of these ways, though the system's designers did not plan it. In part, the complexity of both the EA document itself and its automated system have led to the longevity and stability of the related communities. The creation and maintenance of the social group built around EA's is not so much a matter of sharing common interests, but rather a matter of survival. An engineer has to know, and new engineers have to be taught, how the EA system works in order to do his or her job. According to Brown

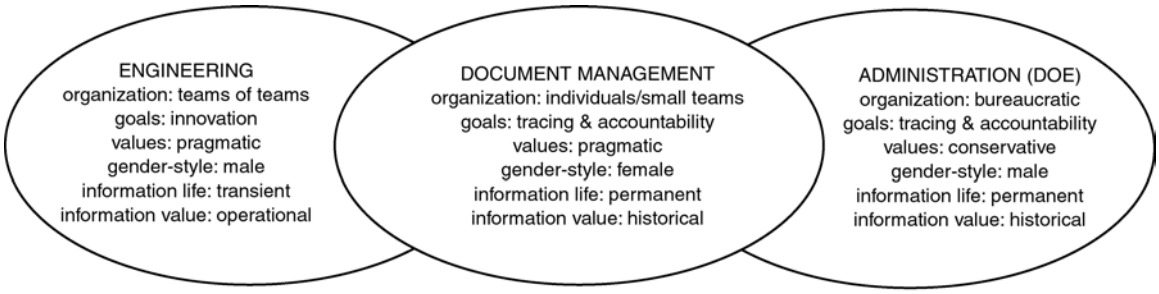


Figure 3

and Duguid, "Strange formats, unexplained generic conventions, jargon, abbreviations, allusions, as well as private languages are all examples of ways in which documents keep people out as much as bring them in." Nowhere is this truer than with EA's. As our interviews progressed we realized that we needed to understand what social and technical channels the engineers use to teach one another and how we could assist with this process.

Bringing new engineers into the system is difficult, largely because so much of the system is implicit. Symptoms of this difficulty include the document manager's recognition that much of their responsibility included training, along with a tendency of many engineering groups to shift responsibility for authoring EAs to designated specialists within their organization.Currently, there is very little available in the way of formal training. When filling out an EA, a new engineer will use a combination of asking an experienced engineer within the project, calling the data management personnel, or perusing their copy of the Technical Business Practices. We are investigating ways in which our system could support training and communication between engineers. One possibility is to provide the capability for electronic annotation among the local community. Much as Brown and Duguid note that conventional forms of publishing broke the link between original documents and marginal commentary, so has the current EA system broken this link. The lack of a draft state and the inability to store informal annotations have probably contributed to the need for numerous change orders over the years.

Another solution, in addition to conventional help screens, FAQ's, improved searches and drop down boxes, is the addition of expert advisors. Through a series of questions, an advisor could lead the new engineer from the selection of the appropriate EA through to its completion. This must be offered as an alternative method for constructing EA's as it would undoubtedly frustrate expert users.

4.4 The Immediacy of Electronic Documents

Another property of the new system that may affect the engineering community is the greater immediacy of electronic documents. Increased immediacy manifests itself in two ways: a web-based system will reduce the time needed to distribute an EA through the national laboratory system from days to moments. Second, it shifts the locus of control (and responsibility) over EAs from document managers to the engineers. Although both of these are clear benefits, we worry that they may reveal hidden flaws or dependencies in our larger systems and business practices. Although these effects are nearly impossible to anticipate, we have found two areas where the increased immediacy of the new system can raise problems.

The first of these is in the approval process. The current system requires that specific *people* (usually a project manager or chief engineer) approve a change to a part. However, the shortened cycle times of the new system raise problems with this focus on individual approvers. For example, it will exacerbate difficulties in locating an appropriate approver: the ability to transmit EAs across country in an instant does engineers little good if the necessary approver is in an all day meeting. This was less of a problem under the old system: delays on the order of a day were not a big issue, and, for longer absences, such as vacations, people informally designated an alternate approver. Not only is the new system more sensitive to delays, but also the shortened times point up the potential flaws in informal delegation of responsibility. For example, a document might wind up in the electronic mailbox of an absent approver, causing delays that might not be noticed by its author until significant time has passed. This was less of a problem with the manual system, since it required people physically interact with approvers to get their signatures. One solution our customers have asked us to consider is the user of formal workflow management with an automated system of roles and delegations to eliminate these bottlenecks. However, we are concerned that this may require excessive changes in the lines of responsibility in the community, and compromise the trust in our system..

The other problem this raises is in controlling who can write an EA against a given part. Under the current system, there are only informal controls: The document managers, who know the engineers and their projects, perform the first check that the EA has a legitimate author. The second check comes from production engineers who, if they were to receive an EA from an unknown person, would call the appropriate designer or engineer for confirmation. Under a fast, automated system, these checks will need to be automated, through password controls on who can create EAs. However, once again, we are concerned about making this as robust as the current human system, and also about interfering ham-handedly in a complex system of social and personal interactions.

4.5 EAs as Historical Records

One of the prominent characteristics of EA's is their stability and permanence. In addition to authorizing specific actions by a manufacturing organization, engineering authorizations also serve as records of an artifact's history. This is particularly challenging in that many of the parts Sandia designs are intended for

military use and must be supported throughout long life cycles on the order of 25+ years. This requires that both design drawings and authorizations be accessible for thirty years or longer. This will not change with the new system, they will be "checked-in" to the system and there they will remain for the next thirty years. These are not only customer requirements: many of them stem from government regulations and are beyond our power to change.

The most significant consequence of this requirement is a severe restriction on the formats we can use for both engineering authorizations and engineering drawings stored in the EBOM system. Sandia engineers have been aggressive in moving from old style drawings to model based mechanical design, embracing CAD tools like Pro/Engineer. However, because commercial software often changes formats, and cannot be assumed to support old formats for the lengths of time we require, our customers required we limit the official record to either text, simple graphics formats or Portable Document Format (PDF). Consequently, although engineers use solid models in their designs, the official record must employ 2-D drawings. This imposes two levels on the representation of designs, and prevents the close integration of working models and drawings of record we would like.

Although the reasons for separating working documents from their historical record are compelling, it raises problems in it least two areas: one is the management of two interacting document contexts and the intertextual issues this raises. The second involves the way the community negotiates the meaning of documents controlling designs and manufacturing processes.

4.6 Intertextual Issues

As mentioned earlier, the EA Manager is a web-based front end to a larger document management system (EBOM) that stores official records of part designs and engineering authorizations. This worked well when engineering used two-dimensional drawings, but raises problems as they move to technologies like model-based design. For example, one of the functions of a model-based design tool is the generation of Numerical Control (NC) programs for machine tools. The accuracy of these programs not only depends on drawing accuracy and correctness; it also depends upon the algorithms used to generate the NC program. One engineer we interviewed cited a case where different CAD tools produced different NC code from the same drawing, due to differences in algorithms used to generate tool paths.

Drawings are inherently less expressive than solid models: a drawing cannot capture everything that is explicit in a solid model, let alone the kind of implicit knowledge illustrated above. The differences in expressive power between the formats required of the historical record and the working models only compounds the innate complexities of intertextual reference. At this time, we have no good solution to this problem. Given the robustness and ingenuity of Sandia's engineering community, we anticipate that our users will develop creative solutions to these as our system receives use. It is essential that we be able to formalize them in subsequent versions of our tool.

4.7 Negotiating Meaning

One of the more interesting aspects of Brown and Duguid's paper is their discussion of the way documents participate in the community's negotiation of their own meaning. This is evident for engineering authorizations in at least two areas: both EAs and drawings of record depend upon other, less rigorously controlled documents and processes for their ultimate meaning. Our discussion of intertextual issues gave examples of this interplay. The second vehicle for negotiating meaning is the formal revision process required for engineering authorizations.

Once an EA or drawing has been released, any changes to it must undergo a formal revision process, requiring new approvals and a new release. Under the current system, however, the process of creating and releasing a new EA is much more complicated than revising an existing authorization. As a result, the engineering community has found it convenient to revise existing EAs whenever possible, simply for convenience. For example, a Complete Engineering Release (CER) authorizes a production agency to begin mass production of a part and releases the part's drawings of record. Many large projects that produce multiple parts find it useful to create one CER and revise it when each new part becomes ready for manufacture, adding that part to a list of released parts. Although this obeys the letter of our engineering practices, it produces documents of excessive complexity and unclear focus. We hope that one of the benefits our tool will provide is to streamline the process of creating new EAs, and discourage this misuse of the revision process. If engineers only revise a document to clarify or qualify its original intent, we believe the revision process will become more directly useful in refining and negotiating the EAs meaning.

Another issue that arises when we look at the negotiation of meaning in an engineering authorization is the role of less formal documents in the process. Because of the rigor demanded of documents in the historical record, many casual notes, e-mails, memos, etc. cannot find their way into the record. An EA must be tied as little as possible to its creation mechanism in case that mechanism is not around in ten or fifteen years.

In spite of these constraints, we hope that EA's will become more fluid: that they will travel faster, allow themselves to be "marked up", and display themselves to many people simultaneously. For example, we are currently exploring the idea of implementing an annotation system. This will allow authors, reviewers and revisors to add informal annotations to the document explaining their reasons for changes, or elaborate on the document's explicit content. The annotations would not be part of the official copy of record, but would be intended as a tool for use by the community of engineers in negotiating and clarifying its meaning. Other people have suggested the annotation system support links to other documents, such as less formal memos or presentations.

Although we are very interested in this idea, it carries a number of pitfalls. The most notable is that current regulations require that any part of an engineering authorization be treated with a fully formal process specified by Sandia's business practices. A narrow interpretation precludes including informal documents in an EA, although we hope an annotation-based approach may someday be allowed

As this discussion suggests, the risks of introducing new software into so complex a community are considerable. This has become our main worry in designing this system. Perhaps the most significant effect of this fieldwork has been to force us to think directly about how we will accommodate the community's

unpredictable response to our system, the new ways people will want to use it, and the new requirements that will emerge from this process. Our main response to this need has been to find a new metaphor, a new way of thinking about the management of engineering authorizations in so rich and dynamic a community. We believe that, based on our fieldwork and our analysis of the social behavior of engineering authorizations, we have found such a metaphor, and that it will be robust enough to accommodate the changes we expect.

5. EA MACHINES VS AGENTS IN A SOCIAL ECOLOGY

Approaching engineering authorizations as social agents revealed properties that a more traditional, documents-as-information-bundles point of view would have left hidden. In particular, it led us to focus on the essential role of informal processes in the handling of engineering authorizations. The most important result of this was a change in our own understanding of our tool. Instead of designing a "machine for managing EAs" metaphor, we have come to think of engineering authorizations, engineers, document managers and our own tool as interacting agents in a complex social ecology.

The central theme of the machine metaphor is that the creation and management of EAs are explicit, deterministic and process constrained. This implies that all significant aspects of EA management can be automated. In contrast, the agent ecology metaphor, by decomposing processes into the loosely coupled interactions of independent agents, brings to the front the interactions between the tool and its human and organizational context.

5.1 The centrality of informal processes

When we began this project, our initial contacts in management and the Technical Business Practice community described engineering authorizations as highly stereotypic in content, form and handling. Although there are well defined requirements for handling EAs, these co-exist with a rich set of implicit, often ad hoc, informal processes that have grown up over nearly 20 years to support our organization's formal requirements. Indeed, we doubt that the EA system would work at all without these informal processes. In addition to the rich community structure described above, three particular issues had immediate, pragmatic impact on our design:

1. The need for informal reviews and sharing. Although Sandia has specified a formal review and approval process for EAs, most of the actual reviewing and correction took place through informal processes (showing a colleague a draft, face-to-face or telephone conversations, etc), so that the formal review process became a largely pro-forma exercise. This meant that anything we did to formalize EA handling should not interfere with this informal process.

2. The indeterminacy of actual processes. We not only discovered that EA management depends upon informal processes, but also that these processes could not, in principle, be formalized. For example, although the process of approving Engineering Authorizations is specified by the organizations involved (DOE & Laboratory management), the requirements are general enough to allow variation across projects and organizations. Determining who should approve a given EA often depends upon implicit knowledge about who was responsible for a given part, or who "holds a stake" in a given system. The process of approving an EA is not ad-hoc: the people involved in the process know exactly what to do and do it consistently. However, this knowledge is informal, contextual and deeply embedded in a complex system of social relationships and practices.

3. The complexity of EA's social participation. As described in the body of this paper, engineering authorization are part of the fabric of a broader community of engineers, managers and document handlers. This fabric of social obligations, assumptions and history serves to insure the correctness and relevance of all authorizations. This is evident in the difference between the "official" role of the document management staff and their actual function in the engineering community. Our biggest problem in designing the EA Manager was to build a tool that would build on this social fabric, rather than rend it apart through ill-conceived formalization and overly simple process standardization.

Although many features of our tool address particular aspects of these problems, all of these are part of a fundamental shift in our understanding of our tool's underlying metaphor.

5.2 Design metaphors: The EA machine vs. an ecology of social agents

Most designers recognize the role of metaphors in user interaction (e.g. the "desktop" metaphor). However, metaphors play a deeper role in defining system functionality, user interaction and even the organization of the project [4]. The initial requirements for our project illustrate a fundamental, underlying machine metaphor for understanding organizations and organizational processes. This was evident in many of our customer's explicitly stated goals: we were asked to standardize the EA process; to achieve quantifiable improvements in efficiency; to eliminate a dependency on specific people in handling EAs; etc. This was also evident in the language our customers used to talk about the tool, with such phrases as "work in the tool", "implement the process", or "click through the stages" figuring repeatedly in our conversations.

In contrast, our fieldwork convinced us that we could best achieve our organizations broader goals of improving the accuracy and traceability of EAs, and of reducing the time and cost to process them by designing a tool that would work within the fabric of the existing engineering authorization community. This led us to pursue a metaphor of engineering authorizations as web-agents: each engineering authorization is an object that "lives" on the corporate network and has both knowledge of organizational requirements for its own content and processing and capabilities to help engineers meet those requirements. However, these agents do not try to control all aspects of the process, but rather to complement and support the underlying human system. For a number of discussions of ecologically inspired approaches to computation see [8].

This metaphor is important, because it allows us to determine the proper functionality of our tool and organize it in a coherent matter. Ways in which the "agent ecology" metaphor differs from the "EA Machine" model include:

5.3 "Working in the tool" vs. "Delegating to the agent"

Most of the people we spoke to in the early stages of the design process talked about creating a tool that EA authors would work "inside" of. This implies that our tool would have to manage all aspects of the EA process: everything of significance would be "in" the tool. Based on our fieldwork, we felt this was neither possible nor desirable. It is interesting to note that, when faced with our findings, proponents of the machine metaphor would dismiss them by saying, "well, we expect you to use common sense." We take this rather vague disclaimer to be an acknowledgement of the breakdown of the machine metaphor (i.e. our inability to automate everything) and the impossibility of stating a solution within the metaphor itself (hence, the vague appeal to common sense).

Because it provides a language for talking about such things as delegation of responsibility between people and the automated system, the social agent metaphor made it easier for us to think about our tool's functionality, the human role and the relationship between them.

For example, when confronted with the problem of determining who should approve a given EA, proponents of the machine model argued for an enforced standardization of these procedures across the laboratory. The unreality of this view is evident in the simple fact that, even though everyone wants standard processes, no one wants to change their own way of doing business. Similarly, no one in upper management is willing to give an order that will disrupt the long standing, successful practices of the labs' individual engineering organizations. In contrast, the agent ecology metaphor helped us find a natural breakdown of responsibility in the approval process: The author of a given EA will explicitly specify who must approve it, based on her knowledge of project roles and responsibilities. The EA "agent" will send itself to those people via e-mail or our corporate workflow tool, for signature. The agent will check the approval using secure authentication techniques and store a record of the approval for future accountability. For the time being, we will not provide roles and delegations as discussed earlier, although we hope that the community practices will evolve to allow this in the future.

5.4 Strong process models vs. states and events

The "EA machine" metaphor requires explicit specification of the process for handling EAs. In the early stages, some of our customers in management sketched interfaces that had a series of buttons across the bottom, instructing us to "make the author click on each button to tell the system they had completed a required step." Part of this model assumed that the system would know when the requirements of one stage had been satisfied. Although proponents of the machine model acknowledged that there would be activities that could not be so modeled, once again, the machine model told us nothing about how to integrate them with the formal processes.

The "agent ecology" metaphor suggested a *state* and *event* approach to the problem. Each agent knows what state it is in. These states correspond to the official EA requirements: draft, approved, distributed, pending incorporation into a drawing (for ACOs). The author or other appropriate person tells the agent when to change state, although the agent may apply rules to check such requirements as the integrity of the document's content. Once a change of state occurs, the EA may trigger events, causing things to happen, such as notification of interested parties, distribution of the EA to subscribers, etc.

This approach lets us give users a powerful means of informally sharing EAs before submitting them for formal approval. As a web object, each EA is identified by a URL. Consequently, an author can paste it into an e-mail or a document, or print it out to share with a colleague. If time allows, we will be able to give the agent the ability to record comments from a reviewer and return them to the author. In this manner, the EA agent helps the author with the informal process without requiring that we specify what the process is. When the author is satisfied, she can then specify who must approve it and instruct the agent to go and get the approval "signatures."

This approach does not enforce requirements on who should approve a given EA, but relies on the current system of social constraints. If an engineer overlooks a necessary approver, this will become evident when the authorization is distributed, and she will be contacted and asked to correct the problem. Although it does not replace the informal system, it does make it more efficient, using electronic routing, digital signatures and automatic recording of the approval.

5.5 Eliminating people vs. co-evolving with them

Perhaps the most disturbing aspect of the machine metaphor is its support for the goal of eliminating people from the process of handling EAs. This follows inevitably from the terms of the machine metaphor: if an EA producing organization is a machine, then, since computers are the ultimate machines, computers should be able to do it all without needing people. Although the labs are under pressure to save money, we must not forget that staff reductions are only one way to accomplish this. Making people more productive is another**.**

As our field work has shown, Sandia's document managers play an essential role in the community that surrounds the creation, use and handling of engineering authorizations, and much of what they do cannot be automated. This, combined with our own ethical resistance to writing software that replaces human beings, placed us in opposition to the goal of saving money by eliminating positions. Fortunately, both our own work and parallel efforts in updating Sandia's Technical Business Practices have made inroads in changing this perception. Currently, management is planning to include the document management staff in a more long term, phase-in of our system. We believe that this will give document managers time to change their job definitions to accommodate the more automated system, and bring to the front their role as teachers, guarantors of data integrity and human mediators between the engineering community and the surrounding bureaucracy.

The agent ecology view supports this observation, in effect holding that the automated and human systems will co-evolve. It is almost a truism that introducing a new system into a complex community changes the community in unanticipated ways. One of the real dangers of the machine metaphor is that, by assuming we can program the community like we program a computer, it leaves us ill equipped to manage these inevitable changes.

Thinking in terms of the agent ecology metaphor gives us a better tool for thinking about the inevitable changes. The next section discusses many of the specific, architectural ramifications of this metaphor.

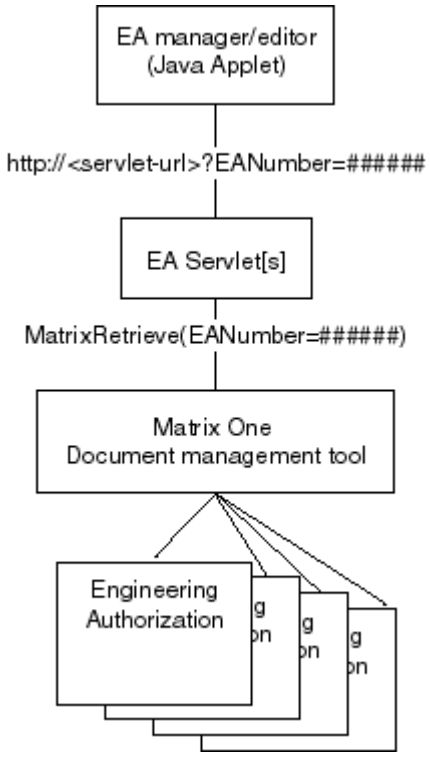


Figure 4

6. ARCHITECTURAL IMPLICATIONS

The "ecology of agents" metaphor did more than influence our understanding of our system's functionality, and give us a vocabulary for talking about its interactions with the community. It also influenced our design of the system software and hardware architecture. Once we changed our goals from the construction of a stand alone tool to making EAs agents on the laboratories intranet, it became necessary to represent EAs as URLs that could be opened in a standard web browser. This would allow EAs to be opened and edited without requiring the installation of special software on the user's machine, to be mailed around using e-mail and to be included in other documents as http links. Figure 4 shows our architecture.

When opening our tool, users see a web page containing a Java applet. We used an applet instead of an html form because it made it easier to provide the powerful user interface functionality we felt was necessary to deliver the agent's capabilities to our users. This includes the ability to run help facilities, advisors and semantic integrity checks efficiently on the client machine, eliminating the time lag required for html form/CGI-based approaches.

The documents themselves are stored in the EBOM database, which is implemented using Matrix One, a commercial document management tool. Matrix One provides such functionality as an object-oriented database, the ability to store documents as part of an object with version control, and a system of events and triggers. In our early designs, we thought of constructing a client program that would run on a user machine and interact directly with Matrix One. This required that the applet communicate with Matrix One using its API (Application Programmer's Interface), a strategy that would have made it hard to treat EAs as objects on the web.

We solved this problem by inserting a layer of Java servlets between the applet and Matrix One. This allows us to represent EAs as simple URLs, with the EA number (its key for retrieval from Matrix One) as a parameter in an http "get". The servlet translates the request into an appropriate form for Matrix One, retrieving the specified EA. The servlet architecture supports other features of the tool, such as access to a corporate human resources database to assist in distribution list management.

A second ramification of the agent metaphor was a greater reliance on events and notifications in handling EAs. One of our requirements was to use a corporate workflow engine for routing EAs. This tool supports notifications, and also has the ability to enforce a required sequence of actions and approvals across a routing process. Based on our agent metaphor, we are using the workflow tool for notification purposes and simple routing, but not using it to enforce a complex approval process. This allows the flexibility we desire in an EA agent.

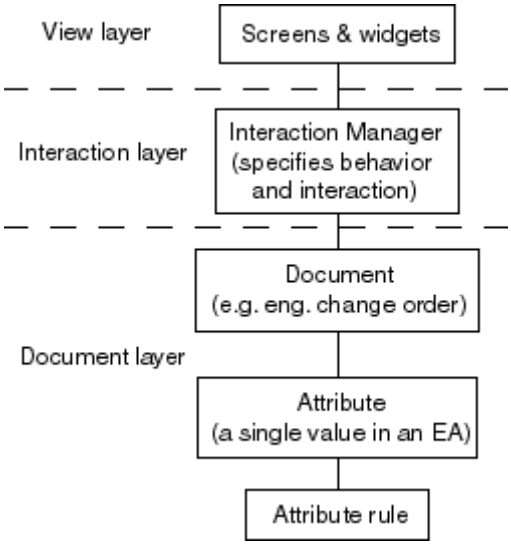


Figure 5

Finally, what is perhaps the most important result of the agent ecology metaphor is that it encouraged us to recognize that our tool would co-evolve with the human community that will contain it. For that reason, we have placed considerable emphasis on making our software easily modifiable. The result is a three-layer object structure (Figure 5). The top layer is the view layer, which consists of screens and screen components. It is concerned only with the screen appearance. The middle layer is the interaction control layer. It determines how the system responds to user requests and the sequence of interactions with the system. Essentially, it determines what the system does. The third

layer is the document layer, which determines the semantics of the document, its integrity requirements, etc. Each document is further divided into attributes, individual information units, and each attribute has zero or more rules for integrity checks.

As a result of this architecture, it is very easy to change anything from the screen appearance, to the agent behavior, to the document's structure and semantics, without changing the larger system. We hope that this will enable our tool to grow and change as the larger human community reconfigures itself around this new participant.

7. CONCLUSION

Our early interest in the social role of information, and in looking at documents as participants in a community has influenced everything from our field-work with users, to our understanding of our system's central interaction metaphor, to the architecture of the software itself. In addition to these specific ideas, our work in tracing the social life of engineering authorizations informed our design efforts in ways both intangible and profound. It has made us more conscious of the need to address organizational and human issues along with our software design efforts. It has made us more sensitive to the political issues surrounding our efforts. Most importantly, it has made us aware of the resiliency and creativity of human communities and their ability to adapt and compensate for the shortcomings of formal systems. Given our own limitations as designers, and the limits of technology itself, we hope our user community will similarly compensate for the shortcomings of our own system.

8. ACKNOWLEDGEMENTS

We would like to thank Sandia National Laboratories indulging our curiosity about work communities. We thank Eric Grose for his role in our field work, and Eunice Young for her insights into the structure of work communities. We also thank members of the EBOM, Web Enabled Engineering Environment, and Product Realization Environment projects for their support. Most of all, we thank Sandia National Laboratories' engineers, business practice authors and document managers for their patience and good humor throughout this process.

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.

9. REFERENCES

[1] Brown, J. S. and Duguid, P. The Social Life of Documents. First Monday: Peer-reviewed Journal on the Internet. 1(1) www.firstmonday.dk.

[2] NWC Technical Business Practice 404: Engineering Authorization System. Sandia National Laboratories. September 1, 1999.

[3] Coyne, Richard. Designing Information Technology in the Postmodern Age: From Method to Metaphor. MIT Press. Cambridge, MA. 1995.

[4] Stubblefield, William A. Patterns of Change in Design Metaphor: A Case Study. *Proceedings: Compter-Human Interaction 98*. Addison-Wesley. Reading, MA. 1998.

[5] Blomberg, Jeanette, Giacomi, Jean, Mosher, Andrea, Swenton-Wall, Pat. Ethnographic Field Methods and Their Relation to Design. In *[7]*

[6] Hutchins, Edwin. Cognition in the Wild. MIT Press. 1994.

[7] Schuler, D. and Namioka, A. Participatory Design: Principles and Practices. Lawrence Erlbaum, Hillsdale, NJ. 1993.

[8] Huberman, B. A. The Ecology of Computation. North-Holland. Amsterdam. 1988