RC22644 (W0211-186) November 19, 2002
Computer Science

IBM Research Report

Who Speaks for Wolf

John C. Thomas, Catalina M. Danis, Alison Lee
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Delhi - Haifa - India - T. J. Watson - Tokyo - Zurich

WHO SPEAKS FOR WOLF?

John Thomas, Catalina Danis, and Alison Lee
jcthomas, danis, alee @us.ibm.com
IBM T. J. Watson Research Center

We are exploring and developing the seeds of a socio-technical Pattern Language as well as associated tools and methodologies to support their use in software design, and developing software components that can be used to develop applications based on the Pattern Language. A Pattern is the named solution to a recurring problem. A Pattern Language is a lattice of such Patterns that cover a coherent field. The socio-technical domain includes groupware, community websites, team ware, and chat. More generally, the socio-technical domain includes a consideration of how technology and social factors interact.

We see many uses for a Pattern Language in this domain; for instance, it can provide support throughout the software development process. It can help developers understand and analyze problems and create designs. Because a Pattern tends to collect together those sub-problems that must be dealt with in concert, a Pattern Language can also serve as a useful guide for project management structure, for testing and debugging, for maintenance, for documentation, and for migration; indeed, it can even help guide marketing and sales efforts by providing a succinct way of relating functionality to benefits. Theoretically, any form of "design rationale" might be useful in these same ways throughout the software life cycle. Inducing developers to write detailed "design rationale" documents de novo, however, has in practice proven extremely difficult. By contrast, once a community of practice becomes familiar with a Pattern Language, it is much easier to simply refer to the utilized Patterns and add a few notations.

Although we feel that our Pattern Language may prove quite useful in software development, its use is certainly not limited to that domain. Such a Pattern Language should also prove useful for groups and communities trying to self-organize and become more effective as well as managers and consultants; anyone who needs to work collaboratively. More broadly, the planet faces problems such as global warming and overpopulation; these require that we find ways to organize collective intelligence and action more urgently than ever before (Thomas, 2001).

We focus on the socio-technical domain partly because it matches our own previous research and experience (e.g., Thomas, 1996; Lee, et. al. 2001; Danis, et. al., 2002) but mainly because we feel this particular domain is timely; considerable knowledge relevant to designing such systems exist, but that knowledge does not yet exist in a form that is very useful or usable for the typical software developer or other practitioner or for communities interested in self-improvement and self-organization.

A Pattern Language can provide useful information for a designer because it focuses on problems in particular contexts and their solutions (Bayle, et. al. 1997). We believe it can also serve as a bridge between the requirements of users and designers, helping to discover many of the unspoken requirements that are otherwise often hard to surface early in design (Thomas and Carroll, 1978). If these requirements are not surfaced, people may be led into finding (and even testing) an optimal solution to a problem that is not the real problem (Thomas and Kellogg, 1989) because it avoids subtle differences in

context, users, or tasks. Moreover, a Pattern Language can also serve as a *lingua franca* among the many stakeholders in complex design situations (Erickson, 2000) where the communication process is not 'simply' one between users on the one hand and designers on the other.

Our Pattern Language is currently organized at the top level with reference to four basic human drives postulated by Lawrence and Nohria (2002). These are the drive to acquire things and experiences, the drive to defend, the drive to bond and the drive to learn. Any given group may be interested in any combination of these, but our working hypothesis is that a designer will typically be trying to focus on one of these issues at a time and this focus can help determine which portion of the Pattern Language to consider. We further break down the drive to acquire into those situations that relate to productivity (i.e., the emphasis is on the product of the activity) and those situations where the greater emphasis is on the pleasure of the process itself. At this point, we've constructed about 75 patterns and are viewing these merely as a seed for a community to further add to as well as discuss the process of building Pattern Languages.

We are attempting to create patterns using several heuristics. One is to deconstruct existing social applications to see what patterns are implicit in their design. A second is to explore mapping Alexander's (Alexander, et. al., 1977) patterns from the physical world to the virtual world. A third heuristic, however, is to look at effective social practices in a variety of cultures. It is in this context that the pattern "Who Speaks for Wolf?" arose.

Problem: Problem solving or design that proceeds down the wrong path can be costly or impossible to correct later. As the inconvenience and cost of a major change in direction mount, cognitive dissonance theory (Festinger, 1957) makes it somewhat likely that the new information will be ignored or devalued so that continuance along the wrong path is likely.

Context: Complex problems such as the construction of new social institutions or the design of complex interactive systems require that a multitude of viewpoints be brought to bear. Unfortunately, this is all too often not the case. One group builds a "solution" for another group without fully understanding the culture, the user needs, the extreme cases, and so on. The result is often a "system" whether technical or social, that creates as many problems as it solves.

The idea for this pattern comes from a Native American (Iroquois) story transcribed by Paula Underwood (1983). In brief, the story goes as follows. The tribe had as one of its members, a man who took it upon himself to learn all that he could about wolves. He became such an expert, that his fellow tribespeople called him "Wolf." While Wolf and several other braves were out on a long hunting expedition, it became clear to the tribe that they would have to move to a new location. After various reconnaisance missions, a new site was selected and the tribe moved.

Shortly thereafter, it became clear that a mistake had been made. The new location was in the middle of the wolves' breeding ground. The wolves were threatening the children and stealing the drying meat. Now, the tribe was faced with a hard decision. Should they move again? Should they post guards all day and night? Or, should they destroy the wolves? And, did they even want to be the sort of people who would kill off

another species for their own convenience?

At last it was decided they would move to yet another new location. But as was their custom, they also asked themselves, “What did we learn from this? How can we prevent making such mistakes in the future.” Someone said, “Well, if Wolf would have been at our first council meeting, he would have prevented this mistake.”

“True enough,” they all agreed. “Therefore, from now on, whenever we meet to make a decision, we shall ask ourselves, ‘Who speaks for Wolf’ to remind us that someone must be capable and delegated to bring to bear the knowledge and interests of any missing stakeholders.”

Forces:

* Gaps in requirements are most cheaply repaired early in development; it is important for this and for reasons of acceptance (as well as ethics!) by all parties that all stakeholders have a say throughout any development or change process.

*Logistical difficulties make the representation of all stakeholder groups at every meeting difficult.

*A new social institution or design will be both better in quality and more easily accepted if all relevant parties have input.

* Once a wrong path is chosen, both social forces and individual cognitive dissonance make it difficult to begin over, change direction or retrace steps.

Solution: Provide automated remindings of stakeholders who are not present. These could be procedural (certain Native Americans always ask, "Who Speaks for Wolf" to remind them) or visual or auditory with technological support.

Examples: Some groups make it a practice to “check in” at the beginning of any meeting to see whether any group members have an issue that they would like to have discussed. In “User Centered Design”, and “Contextual Design” methodologies, an attempt is made to get input from the intended users of the system early on in the design process.

Resulting Context: When every stakeholder’s views are taken into account, the solution will be improved in quality and in addition, there will be less resistance to implementing the solution.

Rationale: Much of the failure of "process re-engineering" can be attributed to the fact that "models" of the "is" process were developed based on some executive's notion of how things were done rather than a study of how they were actually done or asking the people who actually did the work how they were done. A "should be" process was designed to be a more efficient version of the "is" process and then implementation was pushed down on workers. However, since the original "is" model was not based on a very complete picture of reality, the "more efficient" solution often left out vital elements. We hope that this type of mistake is not being remade in the field of knowledge management, but fear that many such systems are attempts to provide a purely technological solution in a situation that calls for a socio-technical approach (Thomas, Kellogg, and Erickson, 2001).

In any complex human endeavor, once a wrong path is initiated, it becomes progressively more difficult to change. For instance, in “A behavioral analysis of the Hobbit-Orcs problem”, (Thomas, 1973), people found it difficult to solve a simple puzzle when it appeared that they had to “undo” progress that has already been made. In more complex endeavors, not only does individual cognitive dissonance make changing direction difficult. In addition, social and logistical factors multiply the difficulties of changing paths.

Technological and sociological "imperialism" provide many additional examples where the input of all the stakeholders is not taken into account. Of course, much of the history of the US government's treatment of the Native Americans was an avoidance of truly including all the stakeholders.

A challenge in applying the "Who Speaks for Wolf" pattern is to judge honestly and correctly whether, indeed, someone does have the knowledge and delegation to "speak for Wolf." If such a person is not present, we may do well to put off design or decision until such a person, or better, "Wolf" can be present.

Known Uses: As a variant of this, a prototype creativity tool has been created. The idea is to have a "board of directors" consisting of famous people. When you have a problem to solve, you are supposed to be reminded of, and think about, how various people would approach this problem. Ask yourself, "What would Einstein have said?" "How would Ghandi have approached this problem?" And so on. This prototype can be viewed at the following:

www.research.ibm.com/knowsoc/

In a previously published study (Desurvire & Thomas, 1993), Human Computer Interaction specialists, computer programmers, and individuals with a background in neither field were asked to do a kind of "heuristic evaluation" of a user interface design under one of two conditions. In one condition, they were simply asked to find as many potential problems as possible and to suggest new features. In a second condition, they were successively asked to try to find potential problems and suggest new features from various perspectives including a cognitive psychologist, a behavioristic psychologist, an occupational therapist, a worried mother, and so on. Subjects in the various conditions had equal amounts of time, but those subjects who were specifically asked to take different perspectives, on average, found more actual usability problems and made a greater number of suggestions than those who were not given the suggestion to look at the problem from these various perspectives.

Discussion. In the case of the tribal experience that gave rise to the story and subsequent heuristic pattern, "Who Speaks for Wolf?", the people of the tribe knew every member of the tribe. That is, all the stakeholders were known, even if all were not present. Furthermore, because of the nature of their culture, it was presumed that each person would have to "live with" the consequences of every other person and their family and friends continuing to be a part of that tribe. Thus, not only were all the stakeholders known, but every stakeholder was likely to remain a stakeholder with continuing influence. Under these circumstances, everyone is wise to consider quite seriously the concerns and perspectives of every other stakeholder, not only because greater overall wisdom will result, but also because of future negative consequences that would occur if someone's concerns were not at least respectfully considered. In many circumstances today, we live under more complex circumstances. During discussions at the DIAC02 conference, it was brought up that we may not always know who all the relevant stakeholders are. Furthermore, because of complex property rights and power relationships, some people may believe that some stakeholders may be ignored without consequence. Arguably, history has repeatedly shown that, in the long term, the belief that some stakeholders may be ignored without consequence, is delusional. Ultimately, those people whose concerns are not addressed because of power relationships will be heard and there will be a reckoning. It is beyond the scope of this paper, however, to present such a case.

What we will address in the remaining paper are strategies for dealing with situations in which a group of people genuinely does want to know who all the relevant stakeholders are but finding out is complex and uncertain. For example, when one of us (JT) worked for a telecommunications company, he worked on a project to automate the re-routing of "intercept calls" via a combination of voice recognition and keypad presses. In this case, we worked closely with the network people, the budget people, the operators, the union, the management and so on. We thought we had thought of every relevant stakeholder and brought them into the process. Then, when we actually went to install the system, the people in charge of the various physical locations ("Branch Offices") where they equipment had to go, refused to allow the systems to be installed unless all of our documentation was reformatted into one that a hardware vendor used (and, therefore one that the maintenance people were familiar with). We agreed to do this and reformatted all our documentation only to discover that the corporate lawyers told us we would not be allowed to publish such documentation because the vendor had copyrighted their format.

Eventually, these issues were resolved, the system was deployed for several years and saved the company money. Clearly, however, we could have avoided a lot of anguish, as well as time delays, if we had extended our circle of stakeholders. In this case, we were simply ignorant of who the complete set of those stakeholders were. There are certainly many similar stories in today's complex society wherein people really do want to include all relevant stakeholders but finding them can be challenging. The following related patterns are meant to offer various suggestions relevant in such situations.

Related Patterns. *Radical Co-location*. In Radical Co-location, all the people working on a project move to a large team room. Providing a space for all stakeholders to be together in one place tends to insure that their input will be given at appropriate times. The more complete pattern is given in the Appendix below.

Advocates. In this pattern, rather than each individual being on the look-out for activities, changes, and situations that may be highly relevant to their concerns (which would be extremely time consuming), a single person (or small group) is charged with the task of understanding the concerns of a related group of individuals and making it their business to watch the environment for situations in which this group should be considered as stakeholders. Examples include public advocates, lobbyists, and activists.

Honeypot. In this pattern, the systems designers, in order to attract the input of all relevant stakeholders, produce an event, or a thing which has extremely wide appeal and wide publicity; e.g., a party, a fair, a website, a free gift. Associated with this is an appeal for input from anyone who has an interest in doing so.

Exploring the Formal Organization. In this pattern, one finds a person or small group of people who know the entire formal organization, or one uses a tool that displays the entire formal organiztion and then explores links to any functions that may be relevant to the situation at hand, contacting those people whose job title, interests, or function may relate. Not all of these will be stakeholders, but clusters who are stakeholders may be revealed with minimal effort or disruption. For example, IBM has a world-wide directory called "Blue Pages" that allows one to search not only by name, but also by job responsibility, project, expertise and interests. This still does not solve the problem that you may not know or recognize the correct term to indicate an important relationship.

Exploring the Informal Organization via Chaining. Here, one begins with people one knows and asks them who they know who would be most likely to be a stakeholder or know a stakeholder relevant to the situation you describe. By working in this recursive fashion, one can connect to relevant stakeholders fairly quickly.

Gaming Scenarios. In this technique, one assigns separate known stakeholder roles to each of several people on the project team. Each is given a different perspective to take and a different set of payoffs. They begin to "play" through a scenario and begin negotiating requirements. In the process of doing this, however, people will be on the look-out for "missing players." Once people begin to confront a concrete (though imagined) reality, other stakeholders who are not being represented become clear.

Hypothesized Anti-Hero. In writing good fiction, it is important, not just to have good heroes, but, perhaps even more important to create interesting, powerful, and believable villains. It is natural in undertaking a project to think of all the benefits of the project and to imagine ourselves as systems designers as the "heroes." Step back and try to imagine that there is someone completely opposed to the system you are designing. What would this person be like? Try to avoid the temptation to imagine them as idiotic, misinformed, or evil (although that can also be a useful exercise). Instead, try to imagine a smart, well-informed, good-intentioned person who is against the project. Why? What is their perspective? What history led to their view?

Multiple Agents. Although so-called "intelligent agents" have not really reached the point where they exhibit intelligence in the human sense, multiple agents can help remind people of sources of knowledge and viewpoints that they might not have otherwise considered.

Public Notice. It might seem too obvious to mention, but of course, wherever feasible, projects whose implications are widespread should be given public notice in appropriate forums; in some, but not all cases, this is a legal requirement. However, the utility of public notice goes beyond the avoidance of a lawsuit. It can result in better design.

Environment Change. Typically, a design is produced for a specific context. One way to test the robustness of design is to imagine a change in the envisioned environment. As a part of that thought experiment, a new set of stakeholders may come to mind as well. Some of these same stakeholders may well be relevant to the project as it stands; they are simply more saliently relevant in the changed environment.

Extreme Characters. Just as stories typically portray the edges of human social and emotional experience, and for that reason we find them fascinating and instructive, we also find extreme characters to be interesting. Djajadinangrat, Gaver, and Frens (2000) from the Royal College of Art have been using a technique during design of imaginging some extreme but very different characters and then building scenarios for how those characters would react to a design; how they might use it; what their concerns might be.

Similar Past Experience. Although it may well be the case that no-one on the design team has experience with exactly the current set of circumstances, it may prove worthwhile to exchange stories of past design projects in which there were forgotten stakeholders; how those stakeholders could have been found earlier. Then, the team can reflect on how that knowledge might apply to the current situation.

Of course, designing any new socio-technical system is a difficult undertaking and none of the methods expressed in these patterns can guarantee that all relevant stakeholders will be discovered ahead of time. Taken together, however, we believe such methods have three important effects. First, the methods increase the chances of finding previously undisclosed stakeholders. Second, the methods generally increase the sensitivity of the design team to potential "side-effects" of their design. Third, such actions demonstrate a values-based design process and if and when new stakeholders are found, negative reactions are more likely to be restricted to material concerns and not include emotional negativity due to being ignored.

Appendix: Radical Co-location Pattern.

Created by John C. Thomas on 5th Sept., 2001

Revised by JCT on June 14, 2002

Synonyms:

"Put the team in one room for the duration of the project", "War rooms"

Abstract:

When small to medium teams of people need to solve a problem or design a novel solution and there are many highly interactive parts, it is useful for the people to work in one large room where people have easy access to each other and shared work objects can be easily viewed, modified, and refered to when necessary.

Problem:

Some problems are amenable to decomposition; that is, the overall problem can be broken down into a series of subproblems and when each of the subproblems is solved, the overall problem will be solved, possibly with slight modification to some of the subsolutions.

In other cases, especially problems that are relatively novel, complex, or "wicked", such decomposition is not possible. If a decomposition is attempted and each of the subproblems is solved, the resulting composition of subsolutions will not be anything close to an overall solution. Under these circumstances, people working alone on their subproblem will become frustrated because all the progress they thought they had made will prove illusory. Morale will suffer. Management will become upset that the apparent progress has not been real and typically attempt a variety of counter-productive measures such as requiring more frequent reports and adding new personnel to meet a schedule.

Context:

In the design of complex systems with many interacting parts, it is often the case that understanding how best to "decompose" a problem cannot be determined ahead of time. Examples include complex software systems, especially where the overall system includes human-human and human-computer interaction, new machinery, novel nuclear power plant designs, complex military operations.

In such a context, handing out separate "assignments" to various individuals or small teams will at first seem to produce progress as each individual or small team carries out their assignment. Unfortunately, when an attempt is made to compose or integrate these subsolutions into an overall solution, the result doesn't work because of unanticipated interactions.

For instance, suppose that a software development team is designing an integrated office support package. Independently, various teams or individuals design various functions. Each of these may be well-designed in itself. However, the combination will be flawed on at least three counts. First, numerous functions will have been duplicated in separate modules. Second, some functionality that would have been useful for the whole package will not have been implemented at all because it would have been too much work for any one team. Third, the user experience will be scattered and inconsistent as separate designers make independent choices about what the user experience will be. In addition, it is quite likely that hard bugs will also be in the design due to the inconsistent treatment of data objects, deadlocks, infinite loops, etc.

There are two main general solutions common in the software development community. First, there may be an attempt to set "ground rules" or "style guides" that everyone is supposed to follow. These will help ameliorate the problem but cannot solve it entirely. Second, there may be overall project meet-ings where people report on progress or even do mutual design reviews. Again, this helps but even if problems are found and resolved, the resolution will require considerable rework.

Forces:

* Most people are naturally gregarious.

* People can concentrate better on difficult mental tasks when it is quiet and when there are a minimum of interruptions.

* Some problems are amenable to decomposition into relatively independent sub-problems; others are not.

* Social cues can be used to guide the interruptability of others.

* Having work-related shared artifacts that can be viewed and understood by others continually leads to productivity.

* Shuffling work artifacts in and out of view in a small space takes time.

* Space costs money and is therefore limited.

* A group will tend to develop useful social conventions when they are co-located.

* Noticing and resolving conflicts among subsolutions early will result in minimizing rework.

* Noticing common problems and solving them collectively as soon as possible will result in maximum efficiency.

* Human performance often shows a "social facilitation" effect; that is, people perform better in the presence of others.

Solution:

When small to medium sized teams work on non-decomposable problems, it is useful for them to be radically co-located in one large room. This room should provide each person some private space and individual work tools (e.g., a computer, a drawing table) as well as numerous spaces for public display of large scale work artifacts (e.g., designs, work plans, diagrams, decisions, group rules, etc.).

Examples:

In the Manhattan Project, people from all over the country were relocated to a relatively remote and isolated area. There they had large workrooms to work on complex problems together.

Recently, automobile companies have empirically compared software work teams that were radically co-located with traditional software development and found the former to be significantly more productive. Interestingly, although before the experience, people thought that they would hate working in a single room, afterwards they said they preferred it (Olson & Olson, 2000).

Resulting Context:

Prior to the experiments at the auto companies, developers were afraid that they would be too distracted by noise and interruptions to get much work done. In fact, social cues can be read fairly well and a potential interrupter can gauge the time to interrupt. In radical co-location, a person might have to wait minutes or hours to resolve an issue by conversation and mutual problem solving. In traditional software development, they may have to wait for a weekly meeting or not discover a problem until integration testing.

People working under conditions of radical co-location tend to develop common vocabulary and artifacts quickly and can easily and efficiently refer to these artifacts. Motivationally, it is also easier to see where the individual's work fits into the larger whole.

Rationale:

In a complex problem solving process, it is most efficient to solve the most difficult constraints first. Similarly, the sooner potential design conflicts or potential design commonalities are discovered, the more efficient the global optimization.

Social groups that work together can rely on subtle cues about whether to interrupt or not. Being alone in the office may seem more conducive to concentration but is still amenable to a knock on the door or a phone call; in this case, the person interrupting generally does not know the state of concentration of the person being interrupted.

When we work separately, it is easy to imagine that others are "slacking off." If we actually see all of our colleagues working, it tends to motivate us to work harder as well.

Related Patterns:

Conversational Support at the Boundaries.

Who Speaks for Wolf?

References:

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl, I. And Agnel, S. (1977) *A Pattern Language*. New York: Oxford University Press.

Bayle, E., Bellamy, R., Casaday, G., Erickson, T., Fincher, S., Grinter, B., Gross, B., Lehder, D., Marmolin, H., Potts, C., Skousen, G. & Thomas, J. (1997) Putting It All Together: Towards a Pattern Language for Interaction Design. Summary Report of the CHI '97 Workshop. SIGCHI Bulletin. New York: ACM.

Danis, C., Lee, A., Karadkar, U., Zhang, J, and Girgensohn. (2002) Getting to Know People: Social Browsing to Support Emerging Community. Manuscript under review.

Desurvire, H. and Thomas, J. (1993). Enhancing the performance of interface evaluators using non-empirical evaluation methods. *Proceedings of the 37th Annual Meeting of the Human Factors and Ergonomics Society,* 1132-1136. Santa Monica, CA: Human Factors and Ergonomic Society.

Djajadiningrat, J., Gaver, W., and Frens, J. (2000). Interaction relabelling and extreme characters: Methods for exploring aesthetic interaction. In *Proceedings of DIS 2000,* 66-71. New York: ACM Press.

Erickson, T. (2000) *Lingua Francas* for Design: Sacred Places and Pattern Languages. In the *Proceedings of DIS 2000*. New York: ACM Press, pp. 357-368.

Festinger, L. (1957) A theory of cognitive dissonance. New York: Harper.

Lawrence, P. and Nohria, N. (2002). *Driven: How Human Nature Shapes our Choices.* New York: Jossey Bass.

Lee, A., Danis, C., Miller, T., and Jung, Y. (2001) Fostering Social Interaction in Online Spaces. In *Proceedings of INTERACT, '01* (Tokyo, Japan, July 2001), IOS Press, 59-66.

Olson, G. and Olson, J. (2000) Distance matters, *Human-Computer Interaction*, **15**, 2-3, 107-137.

Thomas, J. and Carroll, J. (1978). The Psychological Study of Design. *Design Studies*, **1**(1), 5-11.

Thomas, J. and Kellogg, W. (1989). Minimizing Ecological Gaps in Interface Design. *IEEE Software*, pp. 78-86.

Thomas, J. (1996). The Long-Term Social Implications of New Information Technology. In R. Dholakia, N. Mundorf, and N. Dholakia (Eds.), *New Infotainment Technologies in the Home: Demand Side Perspectives*. Hillsdale, NJ: Erlbaum.

Thomas, J. (2001). An HCI Agenda for the Next Millennium: Emergent Global Intelligence. In R. Earnshaw, R. Guedj, A. Van Dam and J. Vince (Eds*.), Frontiers of Human-Centered Computing, Online Communities, and Virtual environments*. London: Springer.

Thomas, J., Kellogg, W.A. and Erickson, T. (2001). The Knowledge Management Puzzle: Human and Social Factors in Knowledge Management. *The IBM Systems Journal*, **40**(4).

Underwood, P. (1983). *Who speaks for Wolf: A Native American Learning Story*. Georgetown TX (now San Anselmo, CA): A Tribe of Two Press.