How Useful Is the Metaphor Component of Agile Methods?
A Preliminary Study

James Tomayko James Herbsleb

June 2003
CMU-CS-03-1523

School of Computer Science
Carnegie Mellon

Also indexed as CMU-ISRI-03-100

Keywords: software engineering methods, eXtreme Programming, metaphor

Abstract

The "metaphor" is the practice of agile processes most ignored by practitioners. A metaphor is meant to be agreed upon by all members of a project as a means of simply explaining the purpose of the project and thus guide the structure of the architecture, thus it is very important for communication, both among the team and with the client. Since both customers and developers alike use the metaphor to clarify the project, a good metaphor should be easily understandable to customers, yet have sufficient content that it can guide architecture development. This paper experiments with the metaphor as a communication tool.

Tutorial Sidebar:

What Is a Metaphor For?

Almost all agile methods explicitly cite "communication" as a key value. This is meant to be communication about the software among the development team, as well as among and with the clients. The use of a metaphor is a powerful tool to achieve this [1].

In eXtreme Programming (XP), probably the most commonly used agile method, there is a practice involving developing a metaphor [2]. Most recognize that there are types of metaphors. There is a "naive metaphor," which is a metaphor very close to the actual product. The financial planner product described in this experiment is an example. Also, there is a more complex metaphor.

The metaphor has two purposes. The first is the communication described above. A user ought to have an easier time speaking and giving examples about a "chameleon" than about a window that changes transparency. A second reason is that the metaphor is supposed to contribute to the team's development of a software architecture.

[1] Bipin Indurkhya, *Metaphor and Cognition,* Kluwer, 1992.

[2] Beck, Kent, *Extreme Programming Explained,* Addison-Wesley, 2000.

The "metaphor" is the practice of agile processes most ignored by practitioners (see above). In fact, Kent Beck, originator of eXtreme Programming (XP), recently felt the need to justify the practice in a recent keynote talk, and offered, depending on the outcome of an audience vote, to "just shut up about it" [1]. While such a vote makes for good theater, practitioners would be better advised to base such a decision on evidence of effectiveness. A metaphor is meant to be agreed upon by all members of a project as a means of simply explaining the purpose of the project and thus guide the structure of the architecture [2], thus it is very important for communication, both among the team and with the client. Since both customers and developers alike use the metaphor to clarify the project, a good metaphor should be easily understandable to customers, yet have sufficient content that it can guide architecture development. As an example, "check writer" is a poor metaphor for financial software tools like Quicken, but "financial advisor" reveals more functionality to users and provides more guidance for high-level design.

There are two times where the metaphor can be tested for effectiveness: soon after it is developed, to see if it helps with design and with communication, and when the architecture is finally developed, to see how the metaphor influenced it. This study focuses on the former, later work will examine the latter.

The Use of the Metaphor to Explain Functionality

There are several dimensions along which the usefulness of a metaphor can be evaluated. A good metaphor, for example, might help the developers understand and agree on the functionality of the system they are designing. Metaphors are often used to explain something that is not well understood in terms of something understood better [3,4, 5]. They may also facilitate communication by giving the team, and perhaps the customer as well, a vocabulary for talking about the system being designed.

Each team was asked to develop a metaphor for their software (see attached assignment, Appendix A). They were asked, as part of the assignment, to keep track of all of the metaphors they considered, and the amount of time they spent on the assignment (neither of which would affect their grades). At the end of the course, students were given individual semi-structured interviews to find out if their descriptions of the metaphor matched that of the other members of their team, and the extent to which they were using their metaphor as a means of communicating their project. They were also asked to sketch the architecture. An analysis of how these sketches matched their final team architecture is the subject of the later paper. Here, only the expression of the functionality is studied. The key questions are:

Cost: What was the cost (in time) for generating a metaphor?

Utility: Were the metaphors considered to be useful in the design process? Were they considered to be useful for communication? Were some metaphors more useful than others?

Experimental Context

The experiment was conducted in the context of a software requirements course. There were 27 software engineers, seven architects, and one civil engineer in the course. For the computational architecture majors, this is a required course; for the engineers, it is an elective. Some of the software engineers are taking a program that requires them to work on a project during their entire year of study in school (the Studio project course). In order to leverage more time for the Studio project, groups were formed from members of the Studio teams plus software engineers from another software engineering degree program, plus architects. One team had an additional architect, since it was one that was not from an already-established Studio project. This project was devoted to nanotechnology, and had an architect as a client. All Studio projects have actual clients with real needs, specific deliverables, and regular meetings with the client. The additional architecture project used an on-campus client, but tried to replicate the other characteristics of a Studio project. In half the cases (variable transparency window,

financial data, and Department of Transportation projects), the client is not computer-literate. This gave us a chance to see if metaphors were more effective with either technically sophisticated or technically unsophisticated clients.

Experimental Design

The student teams had an assignment in their requirements engineering class to develop a metaphor for their project, and to track how long it took them to develop one. The teams had two weeks to complete the assignment. (The exact assignment is reproduced in Appendix A.) Each team submitted a written statement of their final metaphor, along with a description of any false starts and the total time they spent on the assignment. The interviews were recorded on paper, and the architecture collected on the back of the sheet. Even though there was no formal instruction in metaphors or how to create them, the teams had little difficulty developing metaphors that seemed, to them and to us, meaningful and appropriate. To our knowledge, there are no suggestions in the agile literature that formal instruction in metaphor is needed, so this procedure seems close to what one would expect in industrial settings.

Several weeks after the assignment was turned in, the authors met with each team member individually to conduct semi-structured interviews. The interviews began with fairly general, open-ended questions about what the team was building, how it would work, and what the main components would be. Next the team member was asked to describe the metaphor developed by the team, with follow-up questions as needed until the interviewer understood the metaphor and how it applied to the project. Finally we asked each interviewee to respond to six statements about the utility of the team's metaphor with respect to coming up with a design, communication among team members, communication with the customer, and whether they recommend metaphor development for future classes. The interviewee was given five possible responses: strongly agree, agree, neutral, disagree, or strongly disagree. The interviewee was shown both the statements and the possible responses.

The six statements were

> The metaphor has been helpful in figuring out the overall design of the program.
> The metaphor has helped the team find a common vocabulary.
> We often use the metaphor in conversations with each other.
> We often use the metaphor in conversations with our customer.
> The metaphor is useful in helping everyone reach agreement about our requirements.
> I recommend that future classes create metaphors for their projects.

Twenty-seven of 35 students (77%) participated in the interviews. Each of the six projects is represented. Twenty-two interviewees are engineers, five are not.

Results

The Metaphors

The metaphors generated by the six project teams are presented in the table below.

Project	Metaphor	Explanation
Wrist camera	Portrait studio	The software has the capability for transferring images from one device (e.g., PDA, PC) to another, and some image processing capabilities. It is much like a portrait studio, where a camera takes a picture, which is developed, retouched, printed, and distributed.
Wrist camera	Cities and Towns	(same assignment as above). Larger, more capable devices are like cities, in which many services are available. Smaller, less capable devices are like small cities, or even villages, where fewer services are available. Transfer of files is like a train moving from one municipality to another.
Variable transparency window	Chameleon	The window will use nanotechnology to vary opaqueness depending on senor readings, e.g., temperature. It can also generate decorative patterns. This is the architecture project.
Financial planner	Human financial planner	The software follows a specific 5-step method for preparing a financial plan. The metaphor is that the program will behave as would a human planner.
Department of Transportation (DOT) web site	TurboTax	Allows people to register cars, transfer titles, and perform other standard DOT functions. Will be driven by a script that asks questions meaningful to users, automatically populate new forms with data, in a way similar to TurboTax.
Ford Motor Company software architecture tool	C compiler	A tool is being developed to combine architectures of components into one major architectural artifact.

Cost: What was the cost (in time) for generating a metaphor?

The amount of time and the number of false starts were directly reported in the assignment. We found some iterations of the metaphor used to refine the original metaphor ideas, not to come up with new ones. For example, one wrist camera team developed a small city, medium-sized city and large city metaphor in two iterations, the first laid out the terminology, the second paired portions of the functionality with each. About a half-hour elapsed. In contrast, the other wrist camera team came up with a portrait studio in 50 minutes of refinement with five related metaphors that fell short on functionality.

The team working on the variable transparency window also had five false starts and spent three hours development time. The DOT team had four false starts over a 75-minute period before deciding on "TurboTax for Online Vehicle Services." The project developing an architecture assembly tool for a large automotive company developed "a C compiler" metaphor after two false starts in 30 minutes.

The final project was a financial advisor, and the team and client quickly developed the "automated financial advisor" metaphor within minutes of the start of a meeting early in the project. This was done prior to the assignment and the metaphor was in use for several weeks. Therefore, this example is dropped from our results on development time, since we have no meaningful data for it.

The five metaphors we count took an average of 73 minutes, two at 30 minutes and one each at 50, 75, and 180 minutes. False starts averaged three among the five teams. This time and iteration result indicates that the investment in metaphor development is small.

Utility: Were the metaphors considered to be useful in the design process? Were they considered to be useful for communication? Were some metaphors more useful than others?

As described above, the students' overall assessment of the utility of the project metaphors was captured on a 5-point Likert scale anchored by "strongly agree" and "strongly disagree."

We performed a repeated measures analysis of variance (ANOVA) on the questionnaire data, with answers to the six questions as repeated measures, and the six metaphors as a between-subjects fixed effect. We modeled main effects for metaphor, for ratings question (i.e., comparing ratings for the six different questions), and for the interaction of these two variables.

The ANOVA showed no difference for the metaphor main effect ($F= 0.21$, $p= 0.9541$). This indicates that the students' ratings were not consistently higher or lower for any of the six metaphors in the study. There was a highly significant difference, on the other hand, in answers to the six questions ($F=8.41$, $p<.0001$). Duncan's Multiple Range Test revealed that only one score was significantly different from the rest - students were more likely to agree with a statement recommending that future classes should create metaphors for their projects (mean rating = 2.29 on a scale where 1 is strongly agree, 3 is neutral, and 5 is strongly agree). The means for the other questions were 2.9 to 3.5, indicating ratings from neutral (3) to slightly disagree (4). Finally, the interaction of metaphor and type of rating was not significant ($F=1.14$, $p=0.3158$). In other words, there is no evidence that different metaphors were more or less helpful in different ways, nor any evidence that metaphors were more effective for either technically-trained or for non-technical clients.[1]

[1] To confirm these results, we performed and additional repeated-measures ANOVA, with technical versus non-technical client as a between-subject factor, and responses to the six statements as a repeated measure.

Another way to test the usefulness of the metaphors is to see the extent to which the metaphor is reflected in the architecture. We had each student indicate through drawings and words on the back of the interview sheet the nature and form of the architecture as they imagine it. The final architecture is not done, but nearly all of the students were able to draw a "preliminary" architecture when requested.

These will be compared later to the final architectures. For this study, we noted the extent to which language and concepts from the metaphor found their way explicitly into the architecture. For one of the metaphors, the human financial planner, we were unable to separate the language of the metaphor from the language of the software - when a student described a particular function, for example, computing a portfolio analysis, it was not possible to separate literal from metaphoric description, since the two were so close. We excluded these three students from this analysis, but included all other cases. Our preliminary results show that 4 persons could not draw an architecture. For the others, we looked for evidence of at least one word or concept either in the drawing itself (including text labels), or in the explanation they gave of the drawing as they described it to us. While 6 showed some evidence of the metaphor, 14 showed no evidence of the metaphor at all.

These results do not provide support for the utility of metaphor for requirements and early design work. While there were a few scores indicating that some people did judge that the metaphors were useful in various ways, these judgments were not systematic in any way that would help us to tease out the properties of useful metaphors. None of the metaphors was, overall, judged more useful that the others. It would appear from these results that the utility of a metaphor has more to do with individuals and their preferences than with properties of the metaphors themselves. All in all, the metaphors were not very useful for any purpose, with scores ranging from neutral to mild disagreement with statements about their utility.

The results from examining the architecture drawings are similarly unsupportive. Only in very few cases did concepts and terminology from the metaphors seep into the anticipated architecture or the students' explanations of their architectures.

Conclusion

The overall result of this study is that metaphors were not very costly, but in general seemed to have very little utility for the average team member. Data from self-report questions asking about a variety of uses of metaphor, as well as the absence of metaphorical terms or concepts in sketches and explanations of architectures all point in the same direction. There did not appear to be much use for any of the metaphors, for aiding in design, for communicating among the team, or for communicating with the customer.

There were no significant effect for type of client ($F=0.16$, $p>.69$), nor for the type of client by statement interaction ($F=1.08$, $p>.37$).

Interestingly, despite the perceived lack of utility, most students recommended that future classes use metaphors. There may be many reasons for that, of course, including the view that since this class had suffered through it others should suffer similarly. But comments during interviews suggest that many thought that metaphors in many cases would be helpful, even though it did not help their group. Perhaps this explains some of the seductiveness of the metaphor idea - even if it didn't help me, I still believe that in general it is a good thing. Perhaps people are persuaded by their experience of the value of metaphors in other contexts.

While this study found no support for the benefits attributed to metaphors, we want to point out that this study has many limitations. It may be that with more training on metaphors, and more effort invested in their construction, the benefits would be greater. It may also be that metaphors must be introduced at a specific time in the development process to be maximally beneficial - it may have been too early or too late in our study. Finally, the students and the projects, though in many respects they resemble industry developers and industry projects, of necessity there are differences. The students generally are younger, less experienced people working in industry. The projects are unavoidably somewhat artificial. However, we see little reason to believe that younger developers are less likely to benefit than more experienced developers. If anything, one might argue that experienced teams of developers, accustomed to working with each other and with customers, and most likely having more knowledge of their domains, are *less* likely to need metaphors than are these student teams. It will require more research on industry populations to answer the question definitively, but in the limited context of this study, the idea does not show much promise.

References

[1] http://oopsla.acm.org/fp/files/spe-metahpor.html

[2] Beck, Kent, *Extreme Programming Explained,* Addison-Wesley, 2000.

[3] Newkirk, James, and Robert C. Martin, *Extreme Programming in Practice,* Addison-Wesley,2001.

[4] Auer, Ken, and Roy Miller, *Extreme Programming Applied,* Addison-Wesley, 2002.

[5] Wake, William C, *Extreme Programming Explored,* Addison-Wesley, 2002.

Appendix A: The Metaphor Assignment

Due: Friday, 1 November 2002

Making a Metaphor

In agile methods, especially eXtreme Programming, a *metaphor* of the project is developed to help guide a team toward a good architecture and a clearer way to discuss the structure of the software with the client.

For more information, read Kent Beck's *Extreme Programming Explained* [Addison-Wesley, 1999, Ch. 10] or look at web links. Basically, an 'automated checkbook' is a poor metaphor for Quicken, as it does much more. An 'automated accountant' is better, as it captures more functionality.

Try to develop a metaphor for your project among your team. Submit the final metaphor and all the drafts, false starts, etc. Also, keep careful track of the time it takes to build the metaphor, and submit that figure also.

Appendix B: The Interview Questionnaire

[The first several questions were designed to see if the students spontaneously used the metaphor to explain their project. In all interviews, there was at least one person, either a note-taker or the interviewer, who was unfamiliar with the projects. The interviewee was asked to give a quick description of what they were building, ostensibly just to provide some background.]

Background
First, could you give us a little background about your project? Can you describe very briefly what is it that your group is building?
What are the main parts?
How it will work?

Metaphor
Would you describe the metaphor your group came up with?

Utility of metaphor
For the next several questions, I'll read you a short statement and ask you to tell me how much you agree or disagree with the statement, from strongly disagree, disagree, neutral, agree, or strongly agree. [Show them a piece of paper with the rating scale on it.]

The metaphor has been helpful in figuring out the overall design of the program.

The metaphor has helped the team find a common vocabulary.

We often use the metaphor in conversations with each other.

We often use the metaphor in conversations with our customer.

The metaphor is useful in helping everyone reach agreement about our requirements.

I recommend that future classes create metaphors for their projects.