

Aikido and Software Engineering

David Socha

Center for Urban Simulation and Policy Analysis

University of Washington

Box 352350, Seattle, WA 98195

01 206.616.6628

socha@cs.washington.edu

ABSTRACT

Aikido is a martial art whose core philosophy is about conflict resolution and taking care of our opponents and ourselves. My experience, and the experience of fellow aikidoists who work in software development, is that the practice and philosophy of aikido enhance our ability to be effective in the workplace. This paper discusses why this may be.

Categories and Subject Descriptors

K.6.1 [Computing Milieux]: Project and People Management – Management techniques. K.6.3 [Computing Milieux]: Software Management – Software Development

General Terms

Management, Design, Human Factors.

Keywords

Aikido, Conflict Resolution.

1. INTRODUCTION

As the Interdisciplinary Software Engineering Framework acknowledges, humans “in the loop” is an important part of software engineering practice and research. Software engineering results are created via complex human systems, and the success or failure of that work is largely determined by the interactions of people, rather than by technical aspects. Thus, if we wish our discipline to be more effective, we should also look outside technical fields and investigate what we can learn from domains that have a successful history with human systems.

Aikido is one such domain that I have found especially useful as a software practitioner and as a teacher. People see aikido as a martial art, which it is, but often fail to see it as centered around conflict resolution, which it also is [2]. This focus is one of the reasons why aikido can contribute to the practice of software engineering.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WISER '04, November 5, 2004, Newport Beach, California, USA.

Copyright 2004 ACM 1-58113-988-8/04/0011...\$5.00.

In my experience the techniques taught in the safety of the dojo enable me to be more effective when doing software engineering in the workplace. The result of years of practice is a new set of distinctions and abilities that enable me to work more effectively with people outside the dojo. This appears to be a common belief among all of the people I have talked with who are both software developers and aikidoists. For instance, one software developer I know said “I have been doing software for 15 year and aikido for 7, and this aikido stuff comes up every day. Team dynamics. Getting people to work with you. Learning the humility to really listen when people come up with things.... Being able to consider other points of view, not just mine. Sort of a randori¹ going on.” [Robert Del Favero, Personal communication.]

This paper describes the distinctions that enable this, places these distinctions in a model of responses to conflict that extends the responses commonly known in our society, shows how this model is realized in software engineering, and suggests areas for future research.

2. AIKIDO IN THE WORKPLACE

Table 1 illustrates a simple case of when I used aikido in the workplace. The left column paraphrases an actual conversation I had with a developer. It reads from top down. The middle column briefly describes what I was doing. The right column elaborates on the aikido moves I used at each point in the conversation.

As highlighted by the bold words in the right column, aikido consists of a series of blending moves and entering moves.

Blending occurs by physically and mentally trying to see your partner’s perspective. In aikido, as your opponent attacks you, you can blend by stepping aside and going just behind your opponent, to a position where you are safe and where you could give them a comfortable hug, or do a lot of damage to them (this is the martial aspect). You are close to them, physically and metaphorically. Once you see where they are going, you can re-orient them to a direction you want them to go. This typically is done by putting them off balance, gathering them in to your center, and then re-directing them. When done well, you do this very gently – “as if your opponent were your three-year-old daughter,” as my teacher often reminds me.

¹ In aikido, a randori is “a form of practice in which a designated aikidoka defends against multiple attackers in quick succession without knowing how they will attack or in what order.” [4]

Table 1. The aikido moves in a typical design conversation between two developers.

| The Conversation | My Responses | The Aikido |
|---|-------------------------------------|--|
| <i>Fred is excited about his design, and ready to start implementing it.</i> | ← See potential conflict | I'm concerned: Fred's energy is moving toward implementation, but I don't trust his design skills. How can I keep his energy high, while assuring that he does a good job? |
| <i>I ask him about his design, ...</i> | ← Get close to it | I notice this energy, and want to make sure that it will leave us both in a better place, so I enter into a conversation with Fred. |
| <i>... so that I can see what his design is and where it will lead.</i> | ← Try to understand | First I blend with him so that (a) I understand his views, and (b) I gain his trust. |
| <i>This involves many probing questions, some small, some large.</i> | ← Probe | I try to conserve his energy and his dignity as we explore his design. As we converse, I try to keep blending with him, but sometimes I have to ask a question (an entry) that addresses something important and thus touches Fred's center. |
| <i>When Fred's answer validates the design, we both feel good as our confidence in the design increases.</i> | ← Explore | When Fred relaxes with the touch, we are basically in agreement, and continue blending . |
| <i>When Fred resists my question, by trying to gloss over it, I know there probably is a hole in his design.</i> | ← Suspect a problem | When Fred tries to gloss over the problem, he is resisting. That indicates a conflict that needs to be resolved. |
| <i>To determine if there is a problem, I ask a series of increasingly detailed and specific questions.</i> | ← Probe more intensely | I need to understand Fred's view to know if this conflict indicates a real problem. This involves a series of progressively more forceful entries , each to the heart of the matter. |
| <i>When I realize there is a problem, I need to persuade Fred of that.</i> | ← Find a problem | There is a problem, so I need to move Fred to a new place of understanding. This means I need to relax and invite him in by agreeing with the parts that he did well. Then I can take his center and re-direct him. |
| <i>When Fred finally agrees there is a problem, ...</i> | ← Re-orient Fred to see the problem | When we again are facing the same direction, with the same view, we return to blending . |
| <i>... we can work together to find a design that works.</i> | ← Collaborate | We work together toward a solution, both blending with the other's ideas. |
| <i>Once again, he is ready to implement something that we both believe will work. And he is happy to not have wasted time implementing a broken design.</i> | ← Let go; release the energy | I release Fred with his renewed energy focused in a place that meets both of our needs. |

Blending moves are used to achieve several purposes. They put you in a position where you can see your partner's view, where they are going, and what may be concerning them. Blending moves put you in a place where you can take care of your partner while being safe from counter-attack. And instead of blocking your opponent's energy, blending moves allow you to conserve your partner's energy through the entire move. When done well, this feels great to both the attacker and the attacked.

This emphasis on taking care of your opponent is one of the apparent paradoxes of aikido, yet it also is one of the reasons why aikido connects so well to other domains in life.

Entering moves, on the other hand, go to the center of the matter. When an entry works, you always take your partner's balance, even if for just an instant. You might physically put them off balance. Or you may surprise them and capture their mental

attention. In either case, you connect about something that is important to both of you.

If you only want to inform them about something, you can touch their center for an instant. This is like making an assertion: "Did you know I could hit you here?"

If you need to move them to a different place or re-orient them, you take and keep their center, which puts them off balance so that they are easy to move.

Any meaningful conversation or interaction consists of a series of blending and entering moves, as demonstrated in Table 1. Entries inform your partner about what is important in this relationship. The entries are the material, the ideas, the facts for co-design, while the blending is the act of co-creation. The goal in aikido, and in collaboration, is to spend the majority of time blending.

3. AIKIDO AND CONFLICT RESOLUTION

Blending and entering also are interesting and enabling additions to the commonly held possibilities for responses to an attack.

The typical American viewpoint is that there are three possible responses to a perceived attack:

1. Fight
2. Flee
3. Freeze

Another possibility, which often is what the attacker wants, is to:

4. Capitulate

Aikido introduces three additional moves you could make:

5. Blend
6. Enter
7. Appreciate

The first three moves (fight, flee, freeze) are legitimate and often useful. Yet they rarely address the underlying cause of the conflict, and they often harm the attacker, the attacked, or both. Yet, this is our society's predominate model of how to respond to an attack.

The fourth move (capitulate) sometimes does solve the problem, such as when the attacked was doing something wrong and stops. However, it usually is not a win-win solution.

The fifth and sixth moves (blend, enter) tend to resolve the conflict by addressing what is important to both parties in a way that does not harm either party. When done well, it often leaves both parties in a better place with a better solution to their original conflict.

The last move (appreciate) is to thank your opponent when they show you a weakness in your strategies or skills – a place in which you probably want to improve. The aikido philosophy is that your opponent is doing you a service by giving you this feedback. So, when you get hit, your first reaction is to thank your opponent. Just think what it would be like if all developers thanked testers who found bugs in their code.

These moves are not simple to learn. That is why aikido training involves responding to hundreds of attacks per hour, over years, under the guidance of skilled aikidoists. Each attack is a relationship. Each attack is a chance to practice taking care of yourself and your attacker. Each attack helps train your body and mind in different, kinder, and more effective ways to respond to attacks. Each attack makes it a bit easier to stay balanced when your colleagues blow up at you in the workplace.

4. THE AIKIDO METAPHOR

Metaphors are very enabling devices, because they give people a simple model and common language for thinking about and talking about complex domains.

Because aikido is all about relationships, it provides a metaphor that is useful in many human domains. To work with other humans is to have relationships.

Interestingly, this metaphor also describes how a developer works with his code. Programming is like doing aikido.

A programmer's goal is to change the code to meet his or her needs. Programmers also want to make the code better, so they

can more easily meet their next need (and show off to their colleagues). An increasingly common approach is to use test-driven development [1].

Test-driven development happens via a series of entries and blends. First you read the code in order to understand what it is doing, where it is going – this is blending. When you need to know how it *really* works, you write a test – this is an entry. Each test tries to touch a key part of the system. If the test succeeds, the entry became a blend.

If the test fails, your desires conflict with what the code does. You need to resolve this conflict in a way that fulfills your needs and make the code better, leaving you both in a better place. You usually do this by changing the code to fix the test – hopefully with minimal change, minimal force. When the test passes, you once again are blending.

Sometimes the code could do what you want, but the code is not structured to do so. This often requires you to use more energy to reduce the conflict by refactoring the code until both you and the code have the same orientation.

Furthermore, as you work with the code, you learn more about where the code “wants” to go, and what it “can” do. Sometimes this changes *your* desires or opens new possibilities. In this way, the code influences you, just like your partner's energy influences what you do in aikido. As Brian Marick states, programming is “a dance in which the human and the working code are partners, responding to each other's lead” [3]. He continues “the nature of software both solicits change and is predisposed to make that change successful. Software *can* be soft, given the right techniques – discovered through human striving – and proper trust.” I claim that great programming is soft, just like great aikido is soft, as the programmer uses small amounts of energy to produce impressive results.

In a sense, a session of testing and changing the code is like a practice session in aikido. Each test is doing a technique once. Each time you write a test, you learn more about the code, and about what you can do with it. You program through a series of blends and entries.

5. FUTURE RESEARCH

My data is anecdotal at this stage. It would be better grounded if the following research were done:

- What do aikidoists think about how their aikido practice impacts their effectiveness in software engineering?
- Do teams with aikidoists perform better than comparable teams without aikidoists?
- Is there a correlation between the roles people play on development teams and their experience level with aikido?
- How much aikido practice is necessary to get the benefits in software development?
- Do people change their behavior when these models are introduced to them, or do they need months or years of aikido practice in order to change their ingrained habits?

6. CONCLUSIONS

Aikido is a practice that can enhance the effectiveness of software engineers, project managers, and other software development

professionals. Because conflict resolution is at the heart of the aikido philosophy, aikido practice trains the body and mind to react to perceived attacks in a manner that takes care of the person being attacked, the attacker, and the conflict. Embodying this type of reaction pattern makes it easier for us to do the things that are known to help collaboration. This in turn helps us make effective use of our technical skills. Which is the point of being a software development professional.

7. ACKNOWLEDGMENTS

My thanks to Sensei Kimberly Richardson for her insightful comments on this paper, to Elizabeth Davis for being my intellectual uke, to the aikidoists who shared their views with me and led me to realize that my experiences were not unique, and to the software professionals who have walked with me down this path of learning. This research was supported in part by NSF Grant number EIA-0121326.

8. REFERENCES

- [1] Beck, Kent. *Test-Driven Development: By Example*. Addison-Wesley, 2003.
- [2] Crum, Thomas F. *The Magic of Conflict*. Touchstone, Simon & Schuster, 1987.
- [3] Marick, Brian. "Agile Methods, the Emersonian Worldview, and the Dance of Agency." Position paper for an OOPSLA workshop on commonalities of agile methods. <http://www.visibleworkings.com/papers/agile-methods-and-emerson.html>.
- [4] <http://www.fact-index.com/r/ra/randori.html>