

Extracted from:

Agile Retrospectives

Making Good Teams Great

This PDF file contains pages extracted from Agile Retrospectives, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit <http://www.pragmaticprogrammer.com>.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printer versions; the content is otherwise identical.

Copyright © 2005 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

5.1 Activity: Timeline

Use this to gather data in a longer iteration, release, or project retrospective.

Purpose

Stimulate memories of what happened during the increment of work. Create a picture of the work from many perspectives. Examine assumptions about who did what when. See patterns or when energy levels changed. Use this for “just the facts” or facts and feelings.

Time Needed

Thirty to ninety minutes, depending on the size of the group and the length of the increment of work.

Description

Group members write cards to represent memorable, personally meaningful, or otherwise significant events during the iteration, release, or project and then post them in (roughly) chronological order. The retrospective leader supports the team to discuss the events to understand facts and feelings during the iteration, release, or project.

Steps

1. Set up the activity by saying “We’ll fill in a timeline to create a fuller picture of this iteration/release/project. We want to see it from many perspectives.”
2. Divide the team into small groups, with no more than five in a group. Keep people who worked closely with each other together (affinity groups). It’s better to have two small groups representing one affinity than one big group.

Hand out markers and index cards or sticky notes.

Make sure each person has a marker. Although it sounds schoolmarmish, you do need to remind people to write legibly, so people can read the cards.

3. Describe the process.

Ask people to think back over the iteration/release/project and remember all the memorable, personally meaningful, or significant events and write them down, one per card or sticky note.

Remind the group that the point is to see many perspectives—so they don't arrive at a consensus of what was memorable, meaningful, or significant. If it was any of those to one person, that's enough.

Tell them they have ten minutes for this activity.

If you are color coding (see “Variations”) explain what the colors mean and post a legend.

Remind people to write legibly.

4. Monitor the level of activity as people start talking about events and writing cards. If people haven't started writing cards after half the time has elapsed, remind them to start writing. When the groups have a stack of cards, invite people to start posting them (see Figure 5.1, on page 54).
5. When all the cards are posted, invite the team to walk by the timeline and see what others have posted. It's OK for people to add new cards at this point as they remember more events.
6. Allow a break or take lunch before analyzing the timeline.

Variations

We have collected several variations on the timeline activity. We use index cards, sticky notes, markers and dots in a number of ways to pull out both fact and feelings data. For example: For example:

Color Coding Feelings To gather both facts and feelings, use colors to represent emotional states. For example:

- Blue = sad, mad, bad
- Red = challenged, stalled
- Green = satisfied, successful, energetic
- Yellow = cautious, confused
- Purple = fun, surprise, humor
- Salmon = fatigued, stressed

Color Coding Events Use colors to represent types of events. For example:

- Yellow = technical or technology related
- Pink = people or team related
- Green = organization related

Color Coding Functions Use colors to represent functions. For example:

- Blue = developers
- Pink = customers
- Green = QA and testing
- Yellow = technical writers

Color Coding Themes If the team wants to focus on particular matters, use colors to identify events related to specific themes. For example:

- Yellow = team communication
- Blue = equipment usage
- Pink = relationships with customers
- Green = engineering practices

You can pick your own color scheme based on the cards and sticky notes available to you.

Functional Swim Lanes Draw rows lengthwise along the backdrop for the timeline (assuming you aren't planning to post cards directly on the wall—then use ribbon or tape to demarcate the rows). Make a row for each department or function. That group will place their cards only in that swim lane.

In/Out Swim Lanes Draw one line that divides the backdrop in half lengthwise. Use one half for cards for team events and the other half for participants who were involved in the project but weren't part of the core team.

On/Off Use some special shape to represent the people on the project—stars or people-shaped cutouts are good. Ask people to represent when they started on the project by posting a star/people cutout on

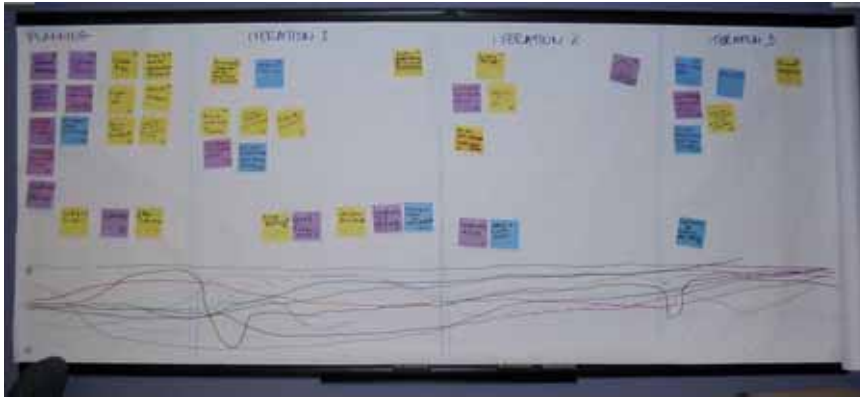


Figure 5.1: A timeline for a retrospective that looked at three iterations. The team was just starting retrospectives and wanted to look back further than just one iteration.

the timeline. Add a star or cutout for people who are no longer on the project or who aren't in the retrospective.

Materials and Preparation

Markers. Index cards or sticky notes. Drafting dots or some other movable tape that allows people to reposition event cards. Painter's tape or tacky stuff to paper the wall.

Backdrop. Cover a long wall with paper to serve as a backdrop. You can overlap flip chart pages or use roll paper. A stretch 6 feet long and 30 inches high is about right for a one-week iteration. For a longer project, you may need 30–60 feet long by 4–6 feet high.

Paper the wall before the retrospective starts.

(For a release or project, prep the timeline with a few time markers such as project milestones, months, or seasons.)

Example

A timeline can display many levels of data about the iteration, release, or project. It can be a simple, chronological listing of events on white index cards. It can also be an extravaganza of data including color-coded themes, cards arranged high or low for meaning, swim lanes for

different functional areas, dots to show positive and negative events, and a space at the bottom with a graph for the ongoing emotional ups and downs. It's easy to get carried away with the possible variations and ask teams to create a timeline with more data than they have time or mental energy to discuss.

When you have only an hour or so for the entire retrospective session, choose a timeline variation that will help to display just enough data. Include both facts and feelings, for sure, but no more than one kind of each. Consult the retrospective goal as a guide for what's most important this time. Keep it simple.

5.2 Activity: Triple Nickels

Use this to gather data or as part of the Decide What to Do phase in an iteration, release, or project retrospective.

Purpose

Generate ideas for actions or recommendations. Uncover important topics about the project history.

Time Needed

Thirty to sixty minutes, depending on the size of the group.

Description

Form small groups. In the groups, each person has five minutes to brainstorm and write down ideas individually. At the end of five minutes, each person passes the paper to the person on his or her right. That person has five minutes to write down ideas that build on the ideas already written on the paper. Repeat until the paper returns to the original writer.

Steps

1. Set up the activity by saying “In this activity, our goal is to generate as many ideas as we can about [topic].” Then explain the process (see the brief description earlier).
2. Divide the team into small groups, with no more than five in a group. Hand out paper for people to write on. Make sure each person has a pen or pencil. Remind people to write legibly so the next person can read the ideas.
3. Describe the process: In the first round, each person will have five minutes to write down ideas related to the topic. Aim for at least five ideas. In subsequent rounds, each person writes ideas that are sparked by the already written ideas or builds on them in some way.
4. Time the group. After five minutes, ring a chime and tell the group to pass the paper to the right.
5. Ask each person to read the ideas listed on the paper.
6. Debrief using these questions:

- What did you notice while you wrote ideas?
- What surprised you? What met your expectations? How?
- What is missing from these lists?
- What ideas and topics should we examine further?

Segue into the next activity where the team will use the ideas generated.

Materials and Preparation:

Paper. Pens or pencils.

Variations

If there are seven or fewer people in the group, don't divide into small groups; do the activity as whole group. Pass the paper only five times.

Examples

With a team of mostly reticent developers and one or two outspoken individuals, an activity like Triple Nickels can allow team members time to think privately yet participate in a process that develops whole-team understanding. It also prevents the few people who are comfortable talking in a group from dominating the discussion. In Triple Nickels, everyone gets the chance to contribute equally to developing the data set, and by the time the data is out, even the more reticent folks usually have something to say about what they wrote or read.



To help the five members of an internal business applications team gather data about their iteration, Aswaria, the retrospective leader, introduced the Triple Nickels activity. She divided the ten people on the team into two groups and passed out paper tablets and pens.

“I'll give you each five minutes to write down five important events that happened during our iteration. Record things you saw or heard during the past fifteen days. Write legibly; make sure someone else can read it.”

At the end of five minutes, she said, “Now pass your papers to the right. Read what you get. You have five minutes to add detail to the items there or add new, related events.”

The team kept passing the papers until each member received their original paper back to read. Some team members laughed at comments; others were shaking their heads. To maintain the theme of “fives,” Aswaria debriefed with questions such as the following: “What five things stand out for you about what you’ve read?” “What five events produced the strongest reactions?” “What are the five most significant events?”

After they finished the discussion, she handed out sticky drafting dots and invited people to post the papers on an area of the wall that she had labeled “Iteration History.”

5.7 Activity: Team Radar

Use this to gather data in an iteration, release, or project retrospective.

Purpose

Help the team gauge how well they are doing on a variety of measures, such as, engineering practices, team values, or other processes.

Time Needed

Fifteen to twenty minutes.

Description

Team members track individual and group ratings for specific factors about process or development practices they want to examine.

Steps

1. Introduce the activity by saying “We agreed on these [fill in the factors] as important to our work. Let’s assess how well we are doing with them, using a scale of 0–10. Zero means not at all, and 10 means as much as possible.”
2. Post the flip chart with the blank radar graph. Ask each team member to approach the chart and place a dot or some other mark that shows their rating for each factor.
3. Lead a short discussion about how the factors affect the work of the team. Consider asking questions such as the following:
 - Where do you see us following these [fill in factors]?
 - Where do you not see us following these [fill in the factors]?

Use the short discussion as a segue to generating insights.

4. Save the completed flip chart graph. Run the activity again two or three iterations later. Compare the two charts as a progress measure.

Materials and Preparation

Flip chart or white board. Markers.

If you know ahead of time what the team will measure using the radar chart, draw the spokes and label them ahead of time (See Figure 5.5

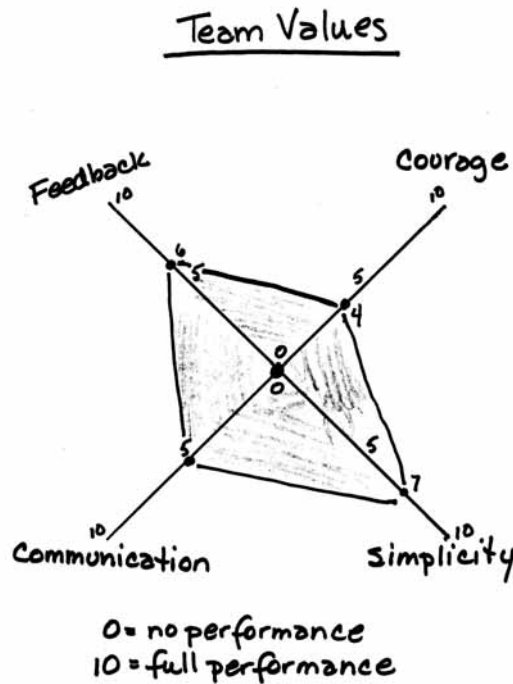


Figure 5.5: This team used the Group Average Radar to gauge how much they were following their team values.

). If the team will brainstorm the measures during the retrospective, draw the radar chart during the retrospective.

Variations

You can use this activity to measure many different factors, such as, engineering practices, team values, working agreements, methods, and so forth.

Group Average Radar This variation is an ongoing measure of progress on a particular measure. Use the radar chart but instead of collecting individual responses, calculate the group average for each measure.

Hand each team member a set of colored cards, one for each factor measured. Ask each person to rate each factor from 0–10 and hand the

card to you. Shuffle the cards (within colors) as you receive them so it's not clear which card came from a particular team member.

Recruit a team member to help with calculating averages. Post the averages on the radar arms. Connect the dots, and color in the area under the line (optional).

Prepare a set of index cards in different colors for each team member. Write the name of one measure on all the cards of a single color. So if you are measuring team values (as in Figure 5.5, on the page before), all the green cards would have "Communication" written on one side, all the blue cards would have "Courage", and so on. Each team member receives a set of cards that includes each factor measured.

Examples

Team Radar is a subjective measure that's intended to generate discussion. This is especially useful when you suspect there's no common definition or criteria to measure against.

For example, one team used a radar to examine how team members perceived their use of a number of engineering practices, including refactoring. One team member rated their refactoring 8; another rated it at 3. During the discussion that followed, it became clear that each had different ideas on when to refactor. Further, the team member who rated her refactoring low was upset with the team member who rated his refactoring high because he was "slacking off by not refactoring enough." By the end of the retrospective, the team arrived at a common definition. Over the next few iterations, the team was more consistent in when they refactored, and resentment faded.

6.2 Activity: Force Field Analysis

Use this in conjunction with an activity that suggests possible changes while generating insights for a release or project retrospective. Use this as part of a planning exercise while deciding what to do.

Purpose

To examine what factors in the organization will support a proposed change and which will inhibit the change.

Time Needed

Forty-five to sixty minutes depending on the complexity of the issue and the size of the group.

Description

The team defines a desired state they want to achieve. Small groups work to identify the factors that could either restrain or drive the change they want. The factors are listed on a poster; then the group assesses the strength of each supporting factor relative to the other supporting factors and repeats the process for inhibiting factors. The team discusses which factors they can influence—either by increasing the strength of a supporting factor or by reducing the strength of an inhibiting factor.

Steps

1. Introduce the activity by saying “If we want this change to succeed, we need to understand more about the factors that will support or inhibit the change.”
2. Describe the process.

Break into small groups (no more than four).

“Each group works for __ minutes to identify factors that will drive or support the change.”

“We’ll do a round-robin report of what you discover and post the results. Then we’ll repeat the process for restraining or inhibiting factors.”

“After we’ve listed both sets of factors, we’ll assess their relative strength and discuss what course of action would be most helpful for implementing the change we want.”

3. Monitor time and the activity level.

While the groups are working, prepare a flip chart like the one pictured in Figure 6.2, on the next page (but don’t fill in the factors yet).

4. When the group is finished with the first task (identifying supporting or driving factors) collect the information the small groups have generated in round-robin fashion, asking for one factor at a time. There’s no need to repeat duplicates; collect only the unique factors.

5. Repeat for restraining or inhibiting factors.

6. Bring the whole group back together, and examine each factor and gauge its strength relative to the other factors. Draw a line toward the center arrow indicating relative strength. Do this first for driving and then for restraining factors.

7. Examine the factors for most effective actions:

- Ask the group how they can strengthen driving factors or mitigate restraining factors.
- Ask the group whether enhancing driving factors or reducing restraining factors is more likely to achieve the desired state.

Materials and Preparation

Flip chart paper or a white board. Markers.

Identify an issue to analyze, perhaps from a list of proposed improvements or another generating insights activity, such as Five Whys or Fishbone.

Example

Force Field Analysis is another tool to ensure that the changes your team identifies in their retrospective actually happen. Combine creating the Force Field Analysis graph with a discussion of influence and control. What can the team directly control to make a change? What can’t they control, and where are their points of influence in the situation? Most teams have more ability to influence conditions than they

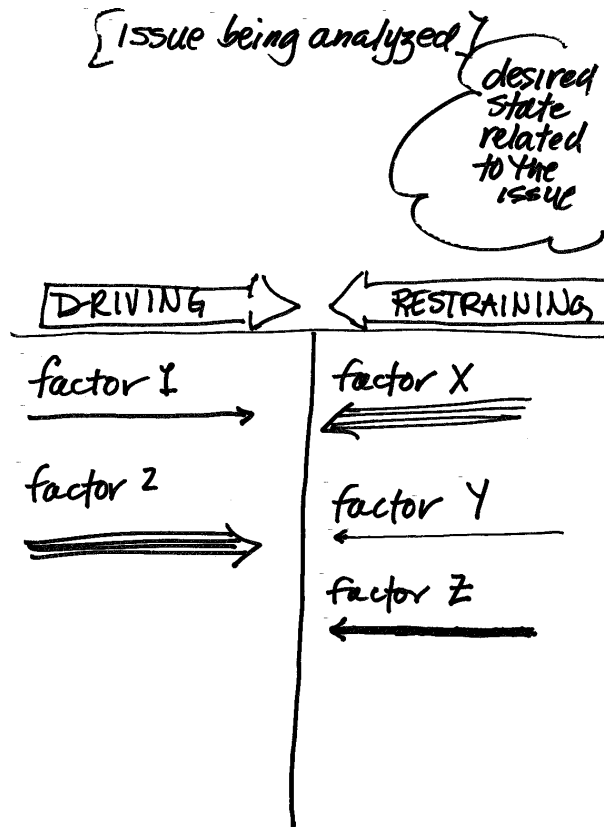


Figure 6.2: Force Field analysis helps the team look at factors affecting a proposed change.

realize; however, a team needs to think about the most effective ways and times to use their influence. Force Field Analysis can help them discern points of greatest leverage and, sometimes, help them see that changing a situation may require more effort than the outcome they desire will be worth. Other times, they may see the forces allied against them and decide to tackle the issue anyway.

One team came into their retrospective wanting to change the way they interacted with the product owner. They were dissatisfied with the limited contact and communication that occurred during the iteration. The product owner answered questions, but only after several days had passed.

Before they analyzed the situation by drawing a Force Field Analysis poster, they understood that the product owner's travel schedule and times of availability were outside their control. Afterward, they also saw they could exert influence best by explaining their concerns to the VP of Marketing, another person with a crazy travel schedule.

They decided that tracking down the VP would take more team effort than they could afford. Instead, they made plans to get the most out of the few product owner contacts available to them.

6.4 Activity: Fishbone

Use this to generate insights in a longer iteration, release, or project retrospective.

Purpose

Look past symptoms to identify root causes related to an issue. Look for reasons behind problems and breakdowns.

Time Needed

Thirty to sixty minutes.

Description

The team identifies factors that are causing or affecting a problem situation and then looks for the most likely causes. After they've identified the most likely causes, they look for ways they can make changes or influence those factors.

Steps

1. Draw a fishbone diagram (see Figure 6.3, on the following page) and write the problem or issue at the fish's head. Include the five W's—What, Who, When, Where, and Why. Label the “bones” of the fish with categories.

Typical categories are as follows:

- Methods, Machines, Materials, Staffing (formerly known as Manpower)
- Place, Procedure, People, Policies
- Surroundings, Suppliers, Systems, Skills

You can use these in any combination, or the team can identify their own categories.

2. Brainstorm factors within each category. Prompt by asking “What are the [fill in a category name here] issues causing or affecting [fill in the problem here.]” Repeat this for each category. Write the issues along the bones, or have people write them on small sticky notes and stick them to the fishbone diagram.

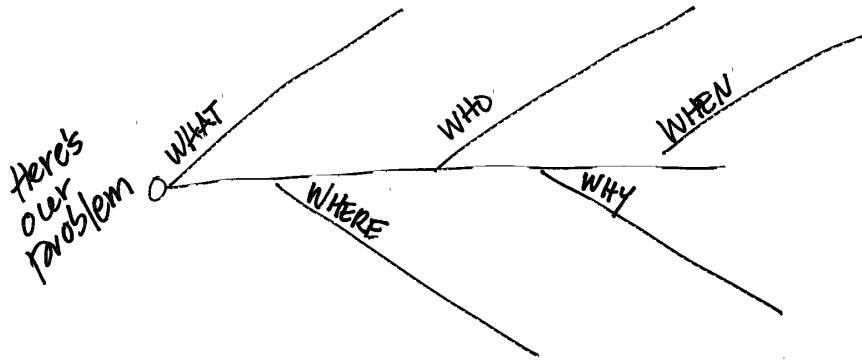


Figure 6.3: Fishbone is a way to look at root causes.

3. Continue asking “Why is this happening?”

Add more branches off the bones as needed.

Stop when the causes are outside the team’s control or influence.

4. Look for items that appear in more than one category. These may be the most likely causes. Engage the group in looking for areas where they can make a difference.

Use the results in the next phase, Decide What to Do.

Materials and Preparation

Markers, sticky notes.

Define the problem statement. Include the five W’s— What, Who, When, Where, and Why—to the extent they are known. Draw the fishbone diagram on a flip or white board. Make a list of the sample categories.

Examples

Use a Fishbone activity to dig into the root causes of a problem, but don’t stop there. A fully branched and labeled diagram is not a deliverable of the retrospective.

If you suspect that a lot of what will come up in the retrospective may be due to issues outside the team’s control, digging into all the problem sources may drain the team’s energy. Choose a different method.

When the issues are more local to the team and under their direct control, the team may be energized by tackling the fishbones.

For example, during a two-week iteration, the build broke five times. The retrospective leader knew the team was frustrated by it, and the broken build would be a prominent topic in the retrospective. He introduced the Fishbone activity with bones labeled “Skills”, “Systems”, “Surroundings”, and “Staffing”.

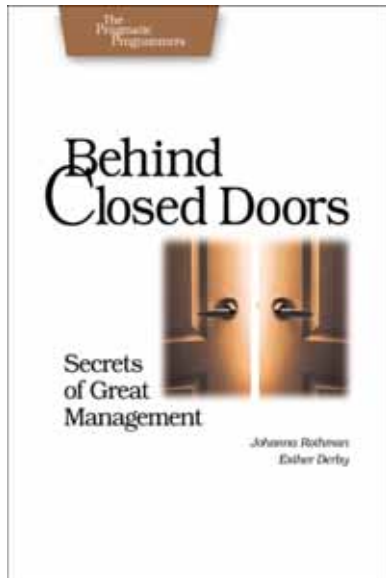
Two or three team members worked in small groups to focus on writing sticky notes for each bone. They covered the “fish” with scales of notes.

When they stepped back to read the notes, they saw two root causes—inexperienced team members working alone (showed up in both Skills and Staffing) and writing new code while waiting for the build to compile (showed up in both Systems and Surroundings). Everyone immediately agreed on a commitment to mentor and pair with newer team members. They identified the second cause as needing more attention and decided to include it as a topic of action planning.

Also by Esther Derby

If you liked this book by Esther Derby and Diana Larsen, you might also enjoy the following title by Johanna Rothman and Esther Derby.

Behind Closed Doors



Secrets of Great Management.

You can learn to be a better manager—even a great manager—with this guide. You'll follow along as Sam, a manager just brought on board, learns the ropes and deals with his new team over the course of his first seven weeks on the job.

From organizing who works on what project when, to helping team members grow and prosper, you'll be there as Sam makes it happen for the entire team.

You'll find powerful tips covering:

- Delegating effectively
- Using feedback and goal-setting
- Developing influence
- Handling one-on-one meetings
- Coaching and mentoring
- Deciding what work to do—and what not to do
- . . . and more.

Full of tips and practical advice on the most important aspects of management, this is one of those books that can make a lasting and immediate impact on your career.

Behind Closed Doors Secrets of Great Management

Johanna Rothman and Esther Derby

(192 pages) ISBN: 0-9766940-2-6. \$24.95

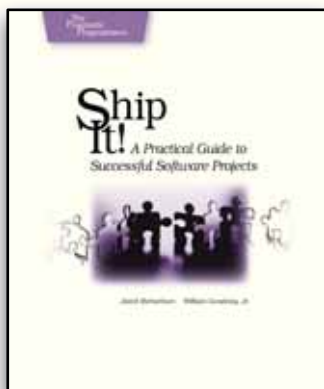
Visit our secure online store: <http://pragmaticprogrammer.com/catalog>

Competitive Edge

We've got you covered—whether it's titles to maintain your competitive edge, or the latest cutting-edge technologies.

For a full list of all of our current titles, as well as announcements of new titles, please visit www.pragmaticprogrammer.com.

Ship It!



Agility for teams. *Ship It!* lets you do just that: on time and on budget, without excuses. You'll see how to implement the common technical infrastructure that every project needs along with well-accepted, easy-to-adopt, best-of-breed practices that really work, as well as common problems and how to solve them.

Ship It!: A Practical Guide to Successful Software Projects

Jared Richardson and Will Gwaltney
(200 pages) ISBN: 0-9745140-4-7. \$29.95

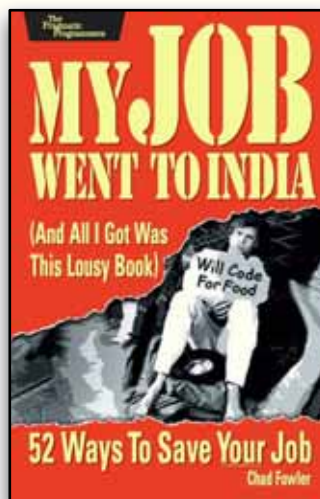
My Job Went to India

World class career advice. The job market is shifting. Your current job may be outsourced, perhaps to India or eastern Europe. But you can save your job and improve your career by following these practical and timely tips. See how to:

- treat your career as a business
- build your own brand as a software developer
- develop a structured plan for keeping your skills up to date
- market yourself to your company and rest of the industry
- keep your job!

My Job Went to India: 52 Ways to Save Your Job

Chad Fowler
(208 pages) ISBN: 0-9766940-1-8. \$19.95



Visit our secure online store: <http://pragmaticprogrammer.com/catalog>

Cutting Edge

Learn how to use the popular Ruby programming language from the Pragmatic Programmers: your definitive source for reference and tutorials on the Ruby language and exciting new application development tools based on Ruby.

The *Facets of Ruby* series includes the definitive guide to Ruby, widely known as the PickAxe book, and *Agile Web Development with Rails*, the first and best guide to the cutting-edge Ruby on Rails application framework.

Programming Ruby (The PickAxe)

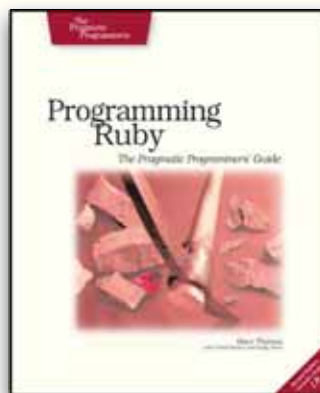
The definitive guide to Ruby programming.

- Up-to-date and expanded for Ruby version 1.8.
- Complete documentation of all the built-in classes, modules, methods, and standard libraries.
- Learn more about Ruby's web tools, unit testing, and programming philosophy.

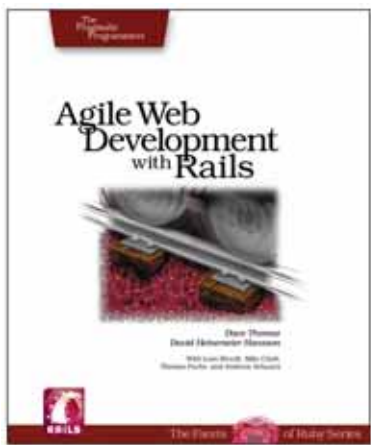
Programming Ruby: The Pragmatic Programmer's Guide, 2nd Edition

Dave Thomas with Chad Fowler
and Andy Hunt

(864 pages) ISBN: 0-9745140-5-5. \$44.95



Agile Web Development with Rails



A new approach to rapid web development.

Develop sophisticated web applications quickly and easily

- Learn the framework of choice for Web 2.0 developers
- Use incremental and iterative development to create the web apps that users want
- Get to go home on time.

Agile Web Development with Rails: A Pragmatic Guide

Dave Thomas and David Heinemeier Hansson
(570 pages) ISBN: 0-9766940-0-X. \$34.95

Visit our secure online store: <http://pragmaticprogrammer.com/catalog>

The Pragmatic Bookshelf

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style, and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help programmers stay on top of their game.

Visit Us Online

Agile Retrospectives's Home Page

pragmaticprogrammer.com/titles/dlret

Source code from this book, errata, and other resources. Come give us feedback, too!

Register for Updates

pragmaticprogrammer.com/updates

Be notified when updates and new books become available.

Join the Community

pragmaticprogrammer.com/community

Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

New and Noteworthy

pragmaticprogrammer.com/news

Check out the latest pragmatic developments in the news.

Buy the Book

If you liked this PDF, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: pragmaticprogrammer.com/titles/dlret.

Contact Us

Phone Orders:	1-800-699-PROG (+1 919 847 3884)
Online Orders:	www.pragmaticprogrammer.com/catalog
Customer Service:	support@pragmaticprogrammer.com
Non-English Versions:	translations@pragmaticprogrammer.com
Pragmatic Teaching:	academic@pragmaticprogrammer.com
Author Proposals:	proposals@pragmaticprogrammer.com