Pattern Mining Patterns

ROBERT S. HANMER, Alcatel-Lucent

Before a software designer can use a pattern the pattern must be mined. One of the most common ways is for the pattern author to interview an expert and document the reusable kernels that perhaps aren't common knowledge. Another way is to assemble a group and use techniques liked brainstorming and guided discussion to extract patterns from the group's knowledge. This paper looks at eight patterns related to extracting pattern information from an expert or a group and writing it as a pattern. It builds on previous work on pattern writing by Meszaros, Doble, Coplien, Rising and others.

Categories and Subject Descriptors: **D.2.11 Software Architecture, I.5.2 Design methodology, K.2 History of Computing, K.3.1 Computer Uses in Education**

General Terms: Pattern Mining

Additional Key Words and Phrases: Software Patterns

1. BACKGROUND

The patterns that are here are part of the larger collection of patterns by many authors on Pattern Mining. These are patterns that I've used repeatedly. They include Patterns for Pattern Writing by Meszaros and Doble (Meszaros and Doble 1996) and Pattern Mining by DeLano (DeLano 1998)

Many of the patterns here are related to building a "language". What I mean in this paper by "language" is a group of patterns that work together to solve a larger problem. They work together by a pattern B resolving the new problems introduced by applying the solution to problem A. For example the patterns from Patterns for Fault Tolerant Software (Hanmer 2007) Riding over Transients and Leaky Bucket Counter work together to solve the problem of dealing with transient faults. Riding Over Transients offers the general strategy but leaves the problem "how do I implement that" which Leaky Bucket counter then solves.

Because many of the patterns here talk about the relationship between patterns, I find it useful to talk about the upper pattern and the lower pattern. An upper pattern will have a lower letter (A<B< <Z) than the lower pattern. The upper pattern is the first pattern that you encounter when using the set of patterns, ultimately it leaves some force unresolved which leads to the need for the lower pattern. Lower patterns resolve issues left by the upper patterns. The lower pattern "D" follows the upper pattern "C". Figure 1 shows this relationship.

$$\begin{tikzcd} C \arrow[dr, bend left] & \\ & D \end{tikzcd}$$

Figure 1. Pattern D follows Pattern C.

1.1 Using Pattern Languages

You build a pattern language for each particular project by pulling patterns from a larger collection of patterns. Using all 23 Design Patterns (Gamma et. al 1996) in one program is an exercise done by

Author's address: 2000 Lucent Lane, Naperville, Il 60563-1337; email: robert.hanmer@alcatel-lucent.com.

many but with limited usefulness in real life. You wouldn't include all of Alexander's A Pattern Language ("APL") (Alexander et. al 1977) patterns in any specific design project; instead you'd pull the patterns that are the most useful in the project's context. Some patterns are going to be included in just about any project's language -- in APL it is Site Repair.

Because the pattern relationships are fluid and specific to each project there is no one canonical mapping between the patterns. A pattern might say that it should be followed with another specific pattern, but that is guidance that might not apply, depending on the situation. An example is that not every situation in which you decide that the system should Ride Over Transients can be implemented by a Leaky Bucket Counter.

While there is no one interconnection between each pattern, there is one linkage in any given situation. In other words if you envision a particular design problem or system that has been built with the patterns you're mining you can focus on the linkages between patterns in that system. That gives you a concrete relationship that you can document in your patterns.

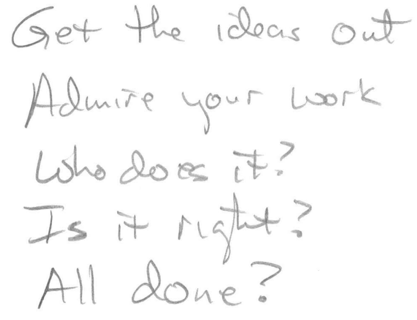
1.2 Pattern Thumbnails

The patterns presented in this paper are summarized in the following table which presents the pattern number and name in addition to a short statement of the intent of the pattern. Each pattern in this table is further labeled with whether it is a pattern about mining patterns or generalized writing of patterns.

2.1. Get the Ideas out	When you're getting started, brainstorm the topics and the pattern names.	Mining
2.2. Connect the Dots	Draw the potential linkages between patterns to see how the patterns and topics tie together.	Mining
2.3. Find What's Missing	Look for places where you haven't identified connections between patterns to see if there are missing patterns.	Mining
2.4. Assign the Work	Divide a large pattern writing group into smaller subsets to actually flesh out patterns.	Mining
2.5. Facilitated Writing	Lead novice pattern writers through an outline to flesh out a pattern.	Writing
2.6. Ideas are Okay	Write down the idea behind a missing pattern even though it isn't a pattern to make sure that you don't lose the thought.	Mining
2.7. Expert Review	Have a subject matter expert review your new pattern to make sure you mined it properly.	Writing
2.8. Admire Your Work	When the pattern is written step back, admire your work and start reviewing it.	Writing
unwritten: Refactor Your Patterns	Revise and restructure your collection of patterns as you learn more about what the building blocks are.	Writing

2. PATTERNS FOR PATTERN MINING

2.1 Get the ideas out



You're writing some patterns about something you know or want to learn. Or you are leading a group that is going to work together to create some patterns.

❖ ❖ ❖

There are so many things that might be patterns, where should you start?

Is it better to start with the biggest concept? Or should the effort start with the smallest concept? Or should you start somewhere in between that will reference the patterns at the extremes?

So many ideas, where to start? Should you try to find the topmost pattern and write that first? Try to find the bottommost patterns that are used in the general solution I'm studying?

Maybe you should google the topic some more -- maybe someone has already written the pattern you're thinking about, in which case you shouldn't waste my time.

Or I could start looking for similar systems that I can use as examples.

Experts are said to be "through the gate" on a topic, meaning that they have internalized and live the patterns of the topic. When they're through the gate it's hard for them to know what the novice (or whoever else you identified as your audience) will want to know and will benefit from.

Therefore,

When just starting singly or in a group just write down a bunch of possible pattern topics. Get the ideas and topics that might be refined into real patterns out on the table in front of you. After that you can begin writing whatever individual pattern you feel most interested in.

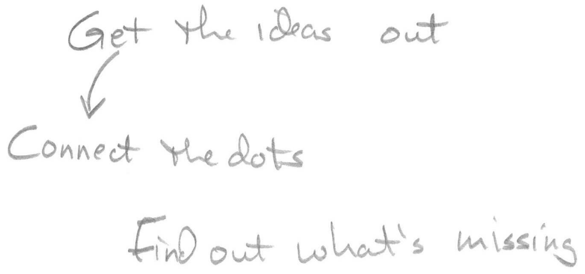
Start by making a list of potential topics for patterns that are related. This is an ordinary brainstorming effort. Don't worry that the topics might not end up as a real pattern -- write them down anyway. If you don't know of a pattern that addresses the topic, or if you can't think of a catchy title, you can write down the short phrase that will remind you of the concept. Since this is brainstorming the idea is to get the idea out of your head and onto paper (or into your word processor). Tools and activities from creative problem solving (reference) can be useful like mindmapping.

❖ ❖ ❖

The point of this exercise is to record the nuclei of potential patterns. Through application of the other patterns in the set that you're reading you'll evolve from a bunch of ideas and topic headings into patterns.

The list of ideas and topics might end up quite large. You'll probably be drawn to the idea or topic to use to start exploring the collection. It might be the one that you have already started writing, or the one that will help you most immediately (like with that field problem you're trying to solve). Then you'll continue digging into the topics by Connecting The Dots (section 2.2) and worry about Finding What's Missing (section 2.3), two patterns suggested by Joseph Yoder.

2.2 Connect the Dots



You're mining patterns in an area. A few patterns in the area are well-known, but you're sure that that there are other patterns too that are incorporated. You've done some initial work on the area by Getting the Ideas Out (Section 2.2), so in addition to the existing patterns you have a list of potential pattern topics.

❖ ❖ ❖

So far you have a list of potential patterns not a cohesive collection of patterns.

Pattern languages are groups of patterns that work together. There are very few problems that aren't trivial that can be resolved by a single pattern.

You're familiar enough with the area you're mining to recognize patterns that are relevant and those that aren't. The patterns on your list that are part of the language you're building will build on each other and work together to create the larger solution.

You might be missing some patterns. The patterns you've identified don't completely cover the domain. There might be patterns in the language that you're building that you haven't discovered yet. By seeing how the patterns that are included in the language relate to each other these unknown ones will start to pop out.

There might be some patterns on your list that aren't really part of the set you're most interested in right now. They can be distracting because you want to include them even though they don't really fit.

Sometimes you're so close to the area that you can't see what the patterns are. This is the case if you're "through the gate". This is a good time to get Expert Review (section 2.7) and ask other experts for their feedback.

Therefore,

Connect the dots. Draw the linkages that exist between the patterns in the example scenario that you know best.

An effective way of connecting the dots is this: Start with one of the patterns -- don't worry where it is in the language -- and think about the problem it solves and how it solves it. The solution undoubtedly has some consequences that are unfavorable for the situation. Ask yourself then what pattern resolves that problem. That is the next pattern in the language and should become your next point of focus. The unfavorable consequences that lead to that next pattern will be forces or context for the pattern. Also look upward in the collection too -- what consequence of a larger pattern does this pattern resolve

❖ ❖ ❖

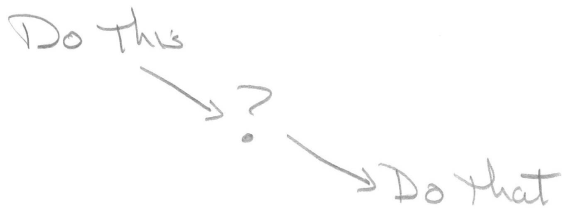
As you connect the dots, you'll need to Find What's Missing (section 2.3) in between the patterns that you started. You'll know that something is missing when you have unresolved consequences or giant leaps from one pattern to another.

You should check that the patterns that you think belong to the set and the linkages really do exist. You should engage the experts for an Expert Review (section 2.7). They can validate that you have identified the right patterns and linkages between them.

You might find that after you connect the dots some of the patterns don't really fit in the collection like you thought they did when you got started. They might have ended up not being referenced by any of the other patterns. That's okay; set them aside for another problem.

Another possibility is that they are too large and grand. Based on the other patterns that you're using in the collection you understand that there are several smaller patterns that should take the place of the larger pattern. Your patterns might also be too small and need to be combined.

2.3 Find What's Missing



The patterns that you're building into a language don't fit together as seamlessly as you initially thought that they would. When you Connected the Dots (Section 2.2) the connection you felt like something was missing.

❖ ❖ ❖

Something's missing. When you read through your connected set of patterns you find pairs of patterns that should flow together but don't.

Patterns are valuable because they identify the tradeoffs needed to resolve the problem. Listing and discussing the forces is one reason they are so useful. The forces in a pattern are either balanced or resolved in the solution or the forces are left unresolved as consequences.

The reason we have multiple patterns that work together to solve a problem is that we use the lower pattern to resolve the unresolved forces of an upper pattern.

If there are forces left unresolved by the upper pattern that are not discussed or used as input to the lower pattern then it will be resolved by some other pattern. If you're confident that you know which pattern follows the one with the unresolved force and that pattern doesn't address the forces there might be another pattern in between.

Does the missing pattern fit between the two that you are considering? Or does it provide an alternate path from the upper pattern?

There are always generic context and generic forces that applies to the whole domain. Some of these aren't worth mentioning because they almost always apply -- for example the von Neumann bottleneck (Backus 1977). If you're writing your patterns to be part of a larger work, say a journal article or PLoP submission, then it will be useful to clearly explain these generic context and forces to help your readers understand why the patterns are shaped the way that they are. As you develop this list of generic constraints you might find that there are new forces.

Therefore,

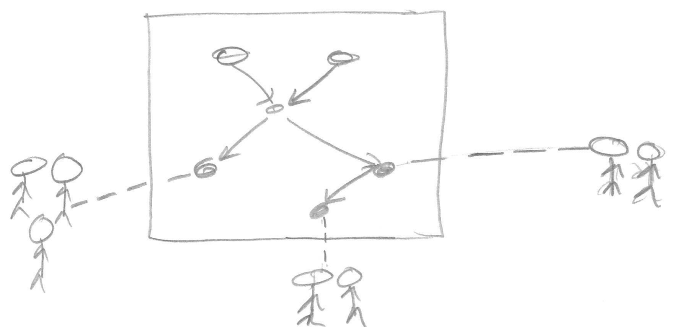
Follow the forces. Forces that disappear or appear as if by magic between two patterns signal that there's a missing pattern -- a pattern that consumes unresolved forces from the upper pattern and that creates unresolved forces for the lower pattern to solve.

What forces of the original larger pattern are unresolved by the lower pattern? What new forces are being resolved in the lower pattern that weren't present or weren't a consequence of the upper pattern? Use these as clues to what pattern is missing. You could use this list of forces as a template for an as yet unnamed pattern: perhaps there is another technique hiding here similar to the first pattern. Sometimes you know that there is a pattern missing, and can identify the forces and consequences, but don't have a name, problem or solution yet. Yet you don't want to or cannot name the pattern yet; in this case remember that during pattern development Ideas are Okay (section 2.6).

❖ ❖ ❖

Sometimes what you find is missing isn't really a pattern. During the initial stages of pattern mining write down the name of the missing concept. You can revisit the concept to look for a pattern later. Another alternative if you identify a concept rather than a pattern is that you should Refactor Your Patterns (unwritten), structuring them slightly differently.

2.4 Assign the work



You're leading a group that is working on a set of new patterns as part of a group. You've Gotten the Ideas (Section 2.1) on paper and you've done some work Connecting the Dots (Section 2.2). Now it's time to Fill in the Gaps (3) and write the patterns.

❖ ❖ ❖

Group writing can be fun, but it can also be frustrating. How can we keep everyone happy and keep forward momentum on the project?

People are all different and unique. It can be challenging to match the pattern writing to the writers. You want to have everyone working at their best so you need to acknowledge these different dynamics.

Some people like working in a group and brainstorming the writing of patterns. Others like to work alone.

Some people are editors and will quickly write something rough and then go back and refine it in an editing process. Some people just put the text down once and never need to revisit and revise it.

Sometimes the patterns are the end goal -- you're documenting a set of patterns to solve some real current or future problem. Sometimes the goal is to build a team that can work together and writing patterns is just one of the group team-building exercises.

Therefore,

Divide the pattern topics up among groupings of the team. Let each of them continue Connecting the Dots (Section 2.2) and Filling in the Gaps (3) in their assigned areas.

The size of the groups should reflect the team dynamics, member's personalities and the overall goals of the exercise. Don't let the groups get too large because then people will not speak up. Having a balance of experts and novices usually works well in a small group. Beware of domineering experts who will intimidate the novices, place them into a group of other experts that feel able to question each other.

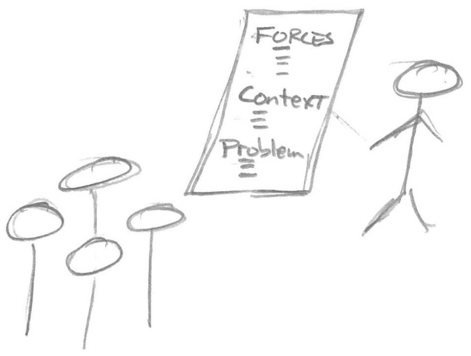
Give the teams clear objectives -- to come back with draft patterns, or to come back with a list of new patterns in the collection. Don't let them leave for their smaller groups with just a vague idea of what is expected from them.

❖ ❖ ❖

Each group can work on its own, or they can all do some sort of Facilitated Writing (5).

Some of the patterns and connections that you derived will end up being discarded -- some totally because they aren't very good and some because they don't fit the collection that you're working on. ..

2.5 Facilitated Writing



Sometimes you have an idea for a pattern and you are trying to get it written by a group. Most of the team is new to patterns and haven't written very many patterns before. You might be mining patterns to give the team the experience rather than with the purpose of developing real patterns.

❖ ❖ ❖

How can you help novice pattern writers write a real pattern?

You could write it yourself but you know from experience that the insights of the others will help you arrive at a better final pattern.

The group probably already knows the solution to the pattern. The hard part of the discussion will be to get them to identify the problem statement.

Therefore,

Sketch out the form of the final pattern on a whiteboard/blackboard/flip chart. Use whatever tool is available to you. Then lead the group to brainstorm to fill in the various parts of the pattern, like the forces, context and problem statement.

This pattern works best with a pattern form that is well structured because the novices can clearly understand each part of the pattern. Best overall is some form of canonical pattern form -- like the Coplien/Indian Hill form (Coplien 1996)

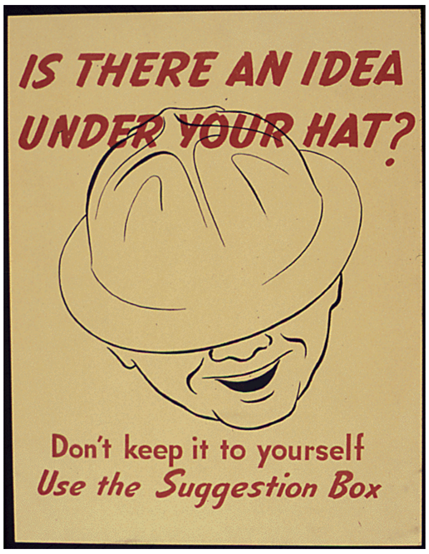
❖ ❖ ❖

Set your expectations realistically for this exercise. You won't end up with a masterpiece; instead you'll probably end up with bullet lists that someone will have to turn into prose. You can Assign the Work (Section 2.4) to get it written or do it yourself. Someone in the group might have developed a passion for taking the notes and turning them into the final pattern.

If someone in the group came up with some particularly insightful and witty comment, be sure to write it verbatim in the notes. That way they'll be able to see their witticism come to life in the final pattern and they'll feel some pride of authorship.

After the group finishes with the pattern ask an expert for the Expert Review (section 2.7) to make sure that the pattern is correct.

2.6 Ideas are Okay



Courtesy US National Archives id NWDNS-179-WP-296

You're writing a collection of patterns. You've Connected the Dots (section 2.2) and Found What's Missing (section 2.3).

❖ ❖ ❖

There's that one place that really doesn't seem like a pattern, but you need to explain a transition from one pattern to another.

Sometimes there's a gap between two patterns that you just can't fill. It's doesn't feel like it's a pattern, but the patterns themselves are too far apart to directly link them via resulting context and context sections. It might be one of the places where a consequence in the upper pattern isn't adequately resolved by the lower pattern, or where there's some totally new element of context that is present in the lower pattern.

You can stretch the patterns so that the resulting context of one does lead directly to the context of the next. This might not feel right if the patterns were well-constructed, easily understood patterns that now have been twisted to fit the collection.

You can leave the gap -- like the old cartoon of the mathematical proof with a cloud labeled "a miracle". This is also unsettling because you know it's not right.

Therefore,

While developing a language, where there's an unexplainable gap write down the idea or concept that is missing between the patterns. Leave it there through at least the first pass of writing. As you sit back and Admire Your Work (section 2.8) and submit the pattern for review in shepherding and a Writers' Workshop you can decide how to handle it going forward. With some time to reflect you might realize that the idea or concept belongs in the other patterns, or you might understand what the missing pattern is.

❖ ❖ ❖

This pattern is about giving yourself permission to put non-pattern ideas into your pattern collection -- at least as a placeholder. In the long term you'll want to handle the idea as either a separate pattern or an extension of the existing pattern but during language development and early reviews it's okay to leave some placeholders. As you flesh out the idea and turn it into a real pattern the Expert Reviews (section 2.7) and Writers' Workshops (Gabriel 2002) will indicate that it is has transitioned from idea to pattern.

2.7 Expert Review

You're writing patterns in an area. You might be mining them yourself or as part of a team. You might be researching the topic and the design from your desk or you might have interviewed an expert to explore the area.

❖ ❖ ❖

Did you write the pattern you thought you wrote?

Sometimes things aren't as they appear. You might have been reading the code unaware that the coding standard in effect at the time of the writing was to place comments after (not before) the line of assembly code that they describe. You might not have heard correctly the comment that the expert made, or when you were mapping it onto what you already knew about the topic you got it wrong.

Therefore,

Have the subject matter expert review your work. This helps you make sure you understand the concepts that you write about in the pattern.

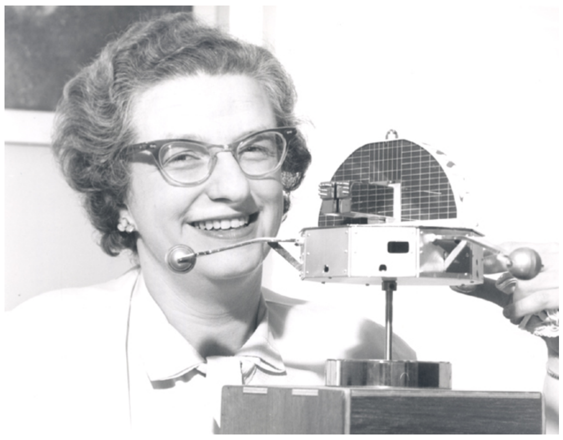
When you are mining patterns by interviewing experts, ask those experts to review your draft patterns. When mining patterns in a group, take the pattern to the domain expert. Mark your pattern as a "draft" until this review (and subsequent revisions) takes place.

❖ ❖ ❖

It's also common to credit the domain expert if you mined the pattern directly from them. A useful way to do this is to write "By <domain expert> as told to <you>".

You should use this as an opportunity to promote patterns too. The Fearless Change collection of patterns has several patterns that apply: Champion Skeptic, Guru on Your Side and Personal touch. (Rising and Manns, 2005)

2.8 Admire your work



Courtesy NASA. GRIN id GPN-2002-000212

Congratulations. You've written a set of patterns that work together to solve some problem that you know something about. You intend the patterns to live on -- in other words they aren't throw-away byproducts of a team-building exercise.

❖ ❖ ❖

What next? You have your set of patterns, what do you do with it?

You could just sit back and admire your work. Or you can try to make it better.

The pattern community uses the Writers' Workshop format to review patterns in a group. They are very effective at letting you hear other people's interpretation of your pattern. (Gabriel 2002)

The risk of letting others read your patterns is that they'll see your faults -- they'll see what you think is your poor writing or poorly conceived patterns. The Writers Workshop is structured to avoid these becoming problems by ensuring that everyone else in the workshop is also a pattern author who's been in the same situation that you will be in.

Therefore,

When the pattern is written step back, admire your work and start reviewing it. Allow yourself a few moments to admire what you've written and to appreciate the effort you've put into the writing. Then think about what you will do with it next.

❖ ❖ ❖

Take it to a Writers' Workshop at PLoP. See which of the patterns you can refine and rewrite to make better. See if any refactoring is needed. Consider other example systems and see if they didn't need some of your patterns, or if they needed other patterns to resolve different problems.

3. ACKNOWLDGEMENTS

I acknowledge Jim Coplien who got me started looking at patterns, first as an "expert" that he mined patterns from and then guiding me as I started writing patterns for my own project. Thanks to Michael Weiss for shepherding these patterns. Thank you to my PLoP 2012 workshop group of Javier Gonzalez-Sanchez, Helen Chavez-Echeagaray, Peter de Waard, Rick Rodin, René Reiners, Kaori Harasawa and Takashi Iba.

(5)

REFERENCES

Alexander, C., S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King and S. Angel, 1977. *A Pattern Language*. *Oxford University Press*: New York, NY.
Backus, J. "Can programming be liberated from the von Neumann style?: a functional style and its algebra of programs," *Comm. of the ACM*, 21:8, 613-641.
Coplien, J. 1996, *Software Patterns: Management Briefings. SIGS Books & Multimedia,* New York.
DeLano, D. E., 1998 "Patterns Mining" in *The Patterns Handbook: Techniques, Strategies and Applications*, Linda Rising, ed. *Cambridge University Press*, Cambridge.
Gabriel, R. P. 2002, *Writers' Workshops & the Work of Making Things: Patterns, Poetry . Pearson Education*, Indianapolis, IN.
Gamma, E., R. Helm, R. Johnson and J. Vlissides, 1996. Design Patterns: Elements of Reusable Object Oriented Software. *Addison Wesley*, Reading, MA.
Hanmer, R., 2007. *Patterns for Fault Tolerant Software. John Wiley & Sons*, Chichester.
Meszaros, G., and J. Doble, 1996, "A Pattern Language for Pattern Writing", http://www.hillside.net/index.php/a-pattern-language-for-pattern-writing.
Rising, L and M. Manns. 2005, *Fearless Change. Addison-Wesley*, Boston.