Short. Smart. Seriously useful.
Free ebooks and reports from O’Reilly at oreil.ly/fr-design

Free ebooks, reports and other articles on UX design, data-informed design, and design for the IoT. Get insights from industry experts and stay current with the latest developments from O’Reilly.
Pair Design
Better Together

Gretchen Anderson and Christopher Noessel
# Table of Contents

**Pair Design: Design Better, Together.**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why Pair Design?</td>
<td>1</td>
</tr>
<tr>
<td>What Is Pair Design, and How Is It Different?</td>
<td>2</td>
</tr>
<tr>
<td>How Does Pair Design Manifest Across User-Centered Design?</td>
<td>6</td>
</tr>
<tr>
<td>What are the Benefits of Pair Design?</td>
<td>11</td>
</tr>
<tr>
<td>Four Case Studies of Pair Design in Practice</td>
<td>15</td>
</tr>
<tr>
<td>Beyond Pairing Two Designers</td>
<td>19</td>
</tr>
<tr>
<td>Where Pair Design Lives and Thrives</td>
<td>25</td>
</tr>
<tr>
<td>Frequently Asked Questions</td>
<td>30</td>
</tr>
<tr>
<td>How Do I Hire for Pair Design?</td>
<td>31</td>
</tr>
<tr>
<td>In Closing</td>
<td>32</td>
</tr>
</tbody>
</table>
Pair Design: 
Design Better, Together

Why Pair Design?

We’ve all been there: it’s time to review a design with some critical stakeholders, you’re prepared, you think you’ve got a great design, and you’re ready to hear feedback and move on to next steps. Except the feedback you’re getting isn’t about refining your great idea. Looking around the table, you realize that your stakeholders aren’t buying your design, because you haven’t addressed a key part of the business need. Or, you hadn’t considered another persona. Or, what you’ve done doesn’t agree with earlier work. Somehow, now that you’re here, you realize that you became lost in the weeds and didn’t focus enough on the big picture. And the part of the picture you did focus on? Well, your pitch isn’t quite landing and the design isn’t the great masterpiece you thought it was.

This happens to a lot of designers, not because they are bad at design, but simply because they are one person—one person working iteratively on a complex problem with a lot of different stakeholders. It’s easy to lose sight of some critical things. One person can fall in love with an idea, and no one is there to point out its rough patches. One person can end up having to do a lot of information management with all of those stakeholders. One person can lose steam and not know where to turn.

This document talks about how pairing two designers can help alleviate these kinds of problems by separating strategic and tactical thinking in regard to a design challenge. Pair design additionally gets to higher quality faster by providing continuous testing of ideas.
before they reach stakeholders, who can see mistakes as failures. And not the kind of failures that stakeholders love.

Pair design can make for better design output, but it also makes for happier designers. Although the craft of design is best practiced solo with some headphones and a great to-do list, it can also be a lonely existence. By working in pairs with partners who share a common language and passion for giving the customer a voice, designers can develop a deeper practice, and build a stronger design culture.

What Is Pair Design, and How Is It Different?

Pair design is the counterintuitive practice of getting more and better UX design done by putting two designers together as thought partners to solve design problems. It’s counterintuitive because you might expect that you could split them up to work in parallel to get double the design done, but for many situations, you’d be wrong. This document will help explain what pair design is, how it works, and tour through the practicalities of implementing it in your practice.

There are several different practices by which pairs of people design interactive systems together. The oldest one we know of is the way pair design is practiced at the small interaction design agency headquartered in San Francisco: Cooper. We each have direct experience with this kind of pair design; there it has been refined across more than two decades’ worth of interaction design, and it shares the most with its development counterpart: pair programming. Thus, we will use it to describe a baseline for the practice. Historically, frog design also paired visual and interaction designers to ensure that the entire experience was cared for. In the later section of this document, we will look at how pair design is practiced in several different contexts today.

There have been lots of different takes on this practice, and when we talk about it around the world, there are two main tenets that can be made quickly, but that are important to establish early and firmly.

Working Together, Closely

The first thing to note about pair design is that it involves two brains on a project at the same time. This doesn’t mean part time, checking in with each other on work that’s been accomplished separately. This
methodology is more properly called a feedback relationship, and even though it's certainly better than nothing, it doesn't achieve the benefits that we'll be discussing. Pair design really means being in the same room, working on the same problem, with both brains focused on the problem simultaneously for the duration of the project. This is central to the practice because it reduces the communication overhead of a design team, allowing higher quality design with less documentation. We'll get more specific on how this plays out across phases of a project in the next section, but for now this is a good place to begin.

Two Defined Stances

The second tenet of the practice is to have clear stances assigned to each designer. These need not be binding forever, as we will see, but each person in the pair takes a distinct and complementary stance toward the design problem as they work together. One generates solutions. That is, one individual materializes solutions to the problem at hand for discussion and iteration. The other synthesizes the proposed solutions. In other words, they identify where the proposal is strong and where it needs improvement, connecting it to the problem statement and the goals of all involved parties; connecting it to decisions that have been made before. Because of these default stances, at Cooper these roles are even named. The former is the generator, or gen, and the latter the synthesizer, or synth.

The generator

In this role, the designer needs to be something of a maker, able to convey an idea clearly and quickly. The key skill is fearless generativity. Throughout most software design projects this means drawing on a whiteboard or digitally: “Here is what I’m proposing for the workflow, or the interactions with the product.” For this reason, gens are generally comfortable drawing and drawing in front of their partner. Additionally, the generator needs to have “fearless generativity,” to be able to come up with a dozen pretty good solutions to a problem even with incomplete information. The person generating ideas needs to be egoless, able to put ideas out that are half baked, get feedback on those half-baked ideas (even to the extent of hearing, “Well, these are half baked.”) and iterating the ideas without taking any of it personally. Gens need to have lots of design patterns in their backpacks, ready to draw on at a moment’s notice, so they
should be familiar with trends and outstanding examples from the field.

**The synthesizer**

In this role, the designer acts as something of an analyst, able to test the design ideas as they are generated and keep the bigger picture in mind about what the design needs to do, what research and feedback has unearthed, and where the team needs to make progress. The key skill is *empathetic skepticism*. Synths give their feedback verbally to the generator, and the two of them discuss and debate the pros and cons on the fly, so they need to be quick on their feet. They also often document design decisions for sharing with the developers and stakeholders: “*Here’s what we recommend for the product and why.*” For these reasons, designers in the synthesizer role need to be skilled at describing designs and explaining rationale in writing. Additionally, the synth needs to have dialectic skills, to keep discussions focused on iterating the designs and not the designers (even to the extent of helping the team fall out love with an idea, helping to see it critically). The role requires the designer to be detail oriented and have a strong memory, to keep the big picture of the system, stakeholders, and users in mind as a reference for designs on the table. Synths need to have lots of first principles, heuristics, and business acumen ready to draw on at a moment’s notice, so they should be familiar with usability, psychology, and business practices.

**Role swaps**

So far, we’ve written as if these roles are rigidly assigned. That makes for good pedagogy, but it can be a little messier in the trenches. Yes, some teams stick to their role from the beginning of the project to the end. But in other teams, the roles shift back and forth. For swapping teams, who does what can be a matter of how they’re feeling in the moment. “OK,” a designer doing generation might say, “I’ve just run out of steam. Do you want to take the pen for a while?” Similarly, a designer doing synthesis might speak up during a pause to say, “Hey, I have an idea that might help us out.”

If the designers are up for it, swapping roles can keep things feeling dynamic and ideas fresh. Some teams might swap roles across projects, acting as a specific role for the duration as needed by their organization. Some teams swap across the course of design sessions.
If you opt to swap, it’s important that each designer knows what role to play in relation to the other. The team must take care to avoid situations in which both are trying to generate at the same time, resulting in a mine-versus-yours comparison in which the loudest voice or the highest-paid-opinion wins. They must also take care to avoid deadlock when no one has an answer to an existing problem. Note that having default roles helps to overcome these problem situations. Additionally, when deadlines approach and the team has less time on hand, each of the pair can focus on his or her responsibilities.

**Aren’t These Just Tasks?**

Can a single designer just take on generating and synthesizing as tasks to be done? Does it need to be two people? Possibly, but it’s not easy. When you come up with an idea, you come up with it for good reasons and are working with a built-in confirmation bias that tells you that your idea is good because it is yours. Synthing your own work fairly takes some superhuman capabilities to overcome that bias. And your experience as an individual might not be as equipped to provide a counterexample as another person with an entirely different set of experiences. So, maybe, kind of, good luck with that?

Can a team simultaneously generate and then swap to simultaneously synthesize its work, to help overcome the confirmation bias? It’s similarly difficult. First, there are the problems mentioned earlier, where the loudest voice or most senior employee wins. Additionally, if someone must cross the threshold from generation to synthesis, and that transition can carry an implication that because someone felt the need to swap, the design must be bad and need fixing. That can raise the defenses of the generator. When it’s one person’s default stance, they’re not being bad by switching to synthesis and criticism. That’s what’s expected because that’s the value they are asked to bring.

Those cautions aside, yes, some individuals and some teams have managed it, but for the reasons discussed, be aware that it’s not a beginner skill.

(You might be interested to know that this issue—whether roles are better if assigned or adopted on an as-needed basis—was probably the biggest point of discussion between the authors and our reviewers.)
ers. So, be aware that there are widely varying opinions on this point.)

**A Note on Colocation**

Modern business practices include team members who might be distributed geographically but collaborate online. Although this can work, it carries risks of participants’ attentions being divided or diluted by weak communications tools and data network slowdowns. Until collaboration technology can catch up, it’s worth noting that pair design works best when the pair can be in the room together for the duration of the project. It gives the team deeply focused attention and the ability to read nuance in the team members’ body language and expressions.

**How Does Pair Design Manifest Across User-Centered Design?**

For the purposes of discussing pair design, we’ll use a four-phase design process: research, analysis, wireframing, and detailed design.

**Research**

As the pair researches the domain, the stakeholders, and users, the pair’s roles are quite similar. Both are there to come to a detailed and actionable understanding of the problems to be solved; the goals to address with the design. Any difference in roles here would be due to other factors.

It’s important to ensure that one person is tasked with taking more literal notes, often acting as a transcriber of what was asked and what was said. That individual can focus on real-time tagging of her notes; such as challenges that were mentioned, pain points that were felt, or workarounds the interviewee has developed. At the same time, you’ll want to see some diagrams and drawings out of your research, as well: an overview of workflows, ecosystems, and relationships. Depending on the specific team members, one person might have more of an affinity to one of these roles over another. If not, being clear about who will do what is an important step.
Analysis and Sensemaking

After completing research, user-centered designers undertake sense-making efforts: that is, formalizing what was learned across research, crafting communications of these findings back to stakeholders, and building the tools that will facilitate design in subsequent phases.

The person synthesizing is tasked with building a big picture in his head in which to frame the design efforts. He will often take a lead in facilitating the sensemaking activities. He might posit some initial questions to frame analysis of the notes, and often takes the lead in drafting personas and scenarios. The person who was focused on literal note-taking will also be a good source for verbatim quotes that illustrate key concepts in users’ actual voices. Gens will reference their notes to participate in making sense of research and co-create the tools needed to make great software happen.

Personas and Scenarios

There are two terms with which you should be familiar. **Personas** are fake characters who embody key characteristics of the people who will be using the software you’re designing. Lots of ink has been spilled during the course of the past two decades explaining and arguing about personas.

**Scenarios** are descriptions of how a persona uses the product or service you are designing to accomplish their goals. They keep teams focused on the big picture of why a user would be engaged with the product and the constraints implied by the context in which they use it. To facilitate easy and inexpensive iteration, scenarios often start as text and only become more refined with pictures across iterations of the design.

If your practice does not entail creating personas or scenarios, it’s OK. It’s just important to understand that the Synth might take the lead in sensemaking efforts after conducting research, which then prompts wireframes and sketching out ideas.
Wireframing and Sketching

It is in wireframing where the roles become more distinct and their differences play out more formally. Wireframing in user-centered design is a catch-all term to describe the process of iterating design ideas at increasing levels of fidelity; for example, from scenarios to loose hand drawings, to sketchy screen layouts, to rendered wireframes. During these activities, the synth will often manage the design activities for the week and structure the tasks for the day.

As the generator’s primary role is to generate design for review by the team, she will lead these activities. For each identified design problem, she draws an initial solution or even a set of possibilities. To do this she stands at a whiteboard (or, more commonly, sits at a digital whiteboard like OneNote) so that the proposed design is immediately visible, as depicted in Figure 1-1.

She talks over her design, voicing her thinking and rationale. The synth responds to the proposed designs, testing and challenging the design against the known requirements and patterns that have already been established in the design system. Often the person synthesizing helps the gen to remain at the right level of fidelity, neither staying too broad nor getting too wrapped up in microinteractions and detail. For example, when working through the overall structure of an app, it’s helpful not to get wrapped up in the layout of every
page. Conversely, it can be helpful to have someone pushing the
design toward details when it might be tempting to keep question-
ing first principles.

As they discuss identified problems, the gen will document key
moments in the conversation, marking up the drawings and making
new ones as needed. Capturing the conversation—including what
was rejected—helps recount the team’s thought process to stake-
holders. Meanwhile, the synth keeps a shared document open to
capture the final design decision and rationale, for sharing with
developers and other team members.

**Detailed Design**

It takes more than interaction design to get a product out the door.
While the pair of interaction designers has been hammering away at
the workflows and interface decisions, other teammates will be
resolving related issues such as industrial, branding, and visual
design. Let us call the activities of all of these coming together in the
final product *detailed design.* (Acknowledging that it is usually a
messier picture than this description presents.)

Efforts in detailed design are often around making the messy
designs-in-progress cleaner and clearer for quick review by stake-
holders but mostly more understandable for sharing with develop-
ers. In this stage, the synth will be in charge of producing the
narrative, whereas the gen will be helping to illustrate with visuals.

Because the gen and synth are doing “heads down” time during this
phase, they might isolate themselves in separate spaces or with head-
phones. When design issues arise they will jump back into the same
room and modes as during wireframing, working through the prob-
lem, and then when the question is answered or the issue resolved,
returning to document the results.

**Other Common Activities**

When we speak of pair design across a process, a few other common
activities often come up: presentations, the creation of working pro-
totypes, and gathering user feedback. For none of these is there a
clear mandate for a strong gen/synth division of labor, but there are
a few notes about how gens and synths handle them having done
other work as a pair.
Presentations

Presentations to stakeholders and developers happen throughout a project, both informally and formally. We feel it’s important that the shared work of the pair can be conveyed by sharing the presentation equally. Neither the gen nor synth is a default presenter. This is not a hard-and-fast rule; and teams might want to adjust this for the seniority and comfort of the pair.

Demos and prototypes

Creating basic prototypes or smoke-and-mirrors demos are becoming an increasingly important part of design exploration, user feedback, documentation, and sometimes assets from which the development team can build. Crafting a prototype is its own kind of generation, but does not necessarily belong to the gen. If both the gen and the synth are working with another team member who is doing the building, they can play the role of synthesizer to the prototypes. Because the creation time of working prototypes can be much more time consuming than the real-time nature of whiteboard wireframing, it does not make sense to have someone synthesizing during the creation phases of a prototype, unless the team can work on it in parallel.

User feedback

Different stakeholders will have different appetites and requirements for how often and thoroughly user feedback can be performed and with what fidelity of prototype. Sometimes, the synth will lead the coordination of these efforts, but it’s important that, like user research, both in the pair conduct the research to build a shared understanding of what has happened and what it means.

Pair design is a way of practicing design that is independent of any particular methodology or set of deliverables. Generally speaking, in design the work tends to become more specialized as the work continues from initial research through to final delivery, looking slightly different depending on the phase of work being undertaken. For paired designers, this means that the roles discussed earlier morph to fit the needs of the project, and having a partner means that each practitioner contributes differently over time. We will look at this dynamic in “Four Case Studies of Pair Design in Practice” on page 15, when we look at pair design in action and in different settings.
What are the Benefits of Pair Design?

Having discussed what Pair Design looks like across some general phases of user-centered design, you might wonder, what does having a close partner get you? What benefits does it convey to the team, to the design, and to the organization to justify the additional resources needed? Unfortunately, we’re not aware of any A/B tests that compare a pair-designing team against others. So we must rely on anecdotal evidence, but this evidence is bolstered by dozens of designers working this way for a decade and being able to compare that to prior ways of working. Here’s what we’ve experienced and heard.

It Makes for Better Design

Pairing improves the quality of the design ideas.

**Pairing forces constant iteration: idea testing and course correction**

By having someone committed to critiquing a shared design, the ideas are continuously challenged and stress-tested by a team who is considering the pros and cons of many approaches to the design problem. By the time the team checks in with stakeholders, it’s rare that an objection can be raised that hasn’t already been considered. This constant testing of ideas also means foundational problems are exposed and caught earlier, saving the team from going too far down a wrong path.

**It brings to bear two brains and two stances**

Pair design is not the same thing as just having two designers to work on a problem. By having someone tasked to generate and someone tasked to synthesize, you gain two vital perspectives on the problem. Someone is looking at it from the bottom up: does this microinteraction make sense in the context of the workflow? Someone is looking at it from the top down: are our design choices consistent with themselves and with the business goals? You have someone thinking of strategy, and someone thinking of tactics. Someone is caught up in the moment and someone is looking at the big picture. These two stances mean the choices must work strategically and tactically at every one of the thousands of interconnected decisions that make up a completed design.
It Makes for Better Designers and Better Design Organizations

Pairing improves the culture of design for teams and organizations.

They are happier

Our clients don’t pay for us just to have fun, but it’s a good thing to ensure designers are bringing their best to the table. Staring at a screen for long solitary stretches can feel dehumanizing and lonely. Pair design makes people happier because of the higher-quality work it produces, the confidence it provides, for the shared successes and shared frustrations, and, yes, even for the human interaction.

This isn’t about introversion or extroversion. Typically, introverts enjoy deep relationships with a few close friends. In a professional design setting, introverts are able to build that deep relationship with a thought partner that doesn’t make it draining. For extroverts, they can enjoy the social nature of the interaction.

We do not mean pairing solves interpersonal problems. There are certainly people who cannot work effectively together, despite best efforts and intentions. But when two people who can work together do pair design, they’re happier than if they had worked in parallel but on separate parts of the problem.

Pair design makes it easier to focus on core aptitudes

Having two roles leaning on different strengths means hiring managers can abandon the quest for the “unicorn designer” who is wise and capable in all ways. It’s much more likely to find people with more focused aptitudes and pair them with another person who possesses complementary skills. If you work with gen–synth pairing, that means you can look for people with a fearless generativity for your generators, and someone well-suited to critical nurturance for your synthesizers. Your gens and synths can then lean into these core aptitudes to do their work. Gens won’t need to worry that they find writing difficult and stressful, and synths don’t need to worry that drawing out their ideas is similarly difficult and stressful.

It’s worth noting in the same breath that some designers fit firmly into a gen or a synth model, whereas others are somewhere in the middle or even more flexible—able to do either. We encourage these
designers to focus on one role at a time. Either for the duration of a project or while in the room with another “swapper.” Both understand the need for the gen–synth dynamic to be progressing the design forward. If two people are “genning,” it risks becoming an ego-driven battle. Likewise, if two people are “synthing,” there is little to respond to and critically nurture.

They cross-pollinate: a mechanism for a learning organization

It's rare that both members of a pair were hired on the same day and are at the same place in their careers. Each will have things to learn from the other, and working together as a community of practice of two will naturally have those effective techniques shared, practiced, and discussed. Then, after a while, the pair can “break up” such that the gen joins another synth and a synth joins another gen, and they will take with them what worked and what didn't work, distributing the best new practices and sloughing off those that no longer work. The cross-pollination in this “square dance” makes for an organization that is continuously sharing and refining its best practices.

Pair Design Makes for a More Effective Process

Pair designers keep design moving forward.

Pairing avoids the problem of dueling whiteboards

When roles are unclear, it’s tempting for designers to respond to one idea with a competing idea. This can give the first idea short shrift. By having only one generator at a time—even if the role swaps back and forth between the pair—it means that an idea’s strengths and weaknesses can be thoroughly explored before considering a competing idea. It also helps remove the designer’s ego from the comparison of ideas. It’s not mine-against-yours, it's ours.

Pair design encourages designers to materialize ideas early

For the synth to have something to respond to, the gen must put something in the world. This pressure means that teams must materialize their ideas quickly and continuously. This means they spend less time in the purely hypothetical, the details of which can be forgotten. The pair leaves a trail of materialized ideas as they iterate toward a buildable solution. This also gives a concrete stem to return to if it turns out a branch was not viable.
Pair design encourages designers to vocalize their rationale

Because the pair begins work in low fidelity and iterates higher and higher, to engage in discussions about the design, the synth still needs to hear a description of what is being drawn and the rationale behind it. This pressures the gen to vocalize her thinking so that the synth can address it as well as what is drawn. This pressures the synth to ask good questions and ensures that nothing is left to assumption. This vocalization also encourages metacognition, such as self-monitoring and self-awareness, which improves the effectiveness of the process.

Pair design encourages constant course-correction

If you’ve ever tried to bring a third party into a deep design problem, you know that the information overhead can be pretty steep. It can take 30 minutes to explain the domain, the strategy, the persona’s goals, the workflow, the step, and the microinteraction before you can get to the crux of the problem. The overhead can discourage feedback, and the longer you go between check-ins, the more you have to explain, and the greater risk the team will have gotten something wrong. By having the pair in the room looking at the same problem at the same time, the information overhead is dropped to zero and the course-correction is continuous, which enables free-flowing, constant, and confident iteration. Meetings with other stakeholders can of course require backing up to earlier design choices, but that the design pair will have lots of practice vocalizing rationale, so this becomes less problematic.

In Short...

Between being happier, producing better work, and being more effective, pair design offers a lot to the designers and the organizations that choose to work this way. People who practice pair design for a while find it difficult and counterproductive to return to working as “genius” designers, doing it all themselves in a slower, fault-prone, and isolated environment.

This is so true that there is a legend at Cooper of one team who found pairing with each other so powerful and fruitful that when they left that company, they sought out opportunities and even interviewed at other organizations as a pair. It really is a powerful shift in how we get great design done.
Four Case Studies of Pair Design in Practice

By now you’ve gleaned a sense of how pair design is different from other ways design is practiced. It involves a deep collaboration between two people concerning the users, the design, the constraints, the stakeholders, and other factors that affect the solution. This collaboration is more than bringing a perspective to a design review or answering questions from a designer. It means dedicated time spent together both developing ideas and testing them out.

Let's looks at a few case studies of pairing in action in both a strategic research/design phase and a more tactical execution phase. Let's begin by looking at a case study from Cooper, that shows some of the above in action.

Cooper: Pair-Designing an App

Christopher Noessel and Suzy Thompson are two designers who have worked as a team for six years and have established roles and methods with which they are quite comfortable. When they designed Kurbo, an app for kids to support healthy eating (see Figure 1-2), Chris played the role of generator on the team while Suzy served as the synth. Because of their long history together, they maintained those roles consistently as they worked together, rather than switching, as we will see in the next case study.

![Figure 1-2. Three screens from the Kurbo app](image)
The Cooper team began the research stage together conducting contextual interviews with families. During this stage, Chris and Suzy swapped responsibilities back and forth, one leading the contextual inquiry sessions, building rapport and listening actively, while the other asked follow-up questions and focused on exhaustive notes. They tended to be intuitive about deriving big patterns and insights quickly and upfront. After each session and in informal discussions about the research, they each shared their impressions; as Chris describes it, “find the big rocks first” in the research, because both people are present for the same sessions, consistently. The details can be then pored through to test hypotheses quickly.

It was during one of these informal sessions that their biggest, hidden challenge emerged—kids don’t make their food choices in a vacuum, their parents, friends, and environment have a big impact on them. This led them into a phase in which they developed scenarios and interaction frameworks for the app. During this stage, the pair used a dedicated room, with Chris running the sketching on a OneNote notebook while mirroring the tablet to a large monitor in the room, and Suzy testing ideas as they go using the personas, scenarios, business goals, and research findings (see Figure 1-2). As they worked through the problems, she led the definition of what’s central to the concept, and what influences it. What happens when a child is 7? 12? 17? What happens if we position this as aspirational instead of shameful?

On a typical day, Chris in turn, sketched out different approaches to these challenges, side by side on his canvas. After they had a set of interaction frameworks that addressed different challenges, Suzy began leading an evaluation of each one against their personas and scenarios, quickly tossing out ideas that didn’t work or scale, identifying unique versus pedestrian solutions. Together they iterated the design to ensure that it met the goals of the personas, their friends and family, and the businesses paying to develop the app.

As the framework solidified, the pair then moved into a more detailed design phase. During this stage, Suzy opened their daily design sessions by reviewing what they had done previously, and identifying the user stories or “problems of the day” to focus on. The pair would run each one through a design/test loop in sketches for several hours in the morning, and often spend afternoons working solo to develop more detailed renderings, prototypes, or develop
enough design rationale documentation for the next scrum meeting with the developer.

With so little information management overhead to deal with, the pair instead put energy into the storytelling of the design for their clients as they did the design, which saved them time and increased the rigor of what they shared with the client. As a review approached, Suzy maintained a list of what states and features were developed, and drove the development of presentation materials. During the presentation, Chris generally led the walkthrough of the scenarios, with Suzy framing the problem and leading facilitation of the client discussion.

The Kurbo app launched in 2015 and won an award at the Interactions16 conference, serving as a good example of how a very structured team can do pair design over the course of a three-month, intense project. The team from Cooper pointed out that the support they received from the organization is a big part of the success of the project and the team.

Across different organizations and industries, design teams and organizations look very different, and it turns out there are many ways to put pair design into practice. To understand how pair design might look differently, let’s look at how other teams practice pair design between designers, and cross-functionally.

**Pivotal Labs: Pair Design with a Client**

As at Cooper, pairing is the status quo at Pivotal. Pivotal was an early adopter of pair programming and pioneered pairing-up designers, and pairing designers with developers. Regularly, two designers on a team are dedicated to a design problem for the duration of a project, and pair programming is a natural mode for engineers. Pivotal makes its own products as well as acting as consultants for clients, and pairs function in either setting. Sometimes, the second designer on the team is a designer from the client’s organization.

When Aaron, a lead designer at Pivotal, began the design session we observed, he introduced Michael as his design partner. Michael sat on a tall stool at a high table, and Aaron stood next to him. They shared a monitor, and each had a mouse and keyboard, much like traditional pair programming, except rather than looking at code, they were looking at a design sketch. At Pivotal Labs, the roles of
generating ideas versus synthesizing information are called, respectively, “Driving” and “Navigating.” In this configuration, the designer using the computer is driving the design session, while the other is navigating—consulting notes, testing ideas, checking edge cases. It’s only at the end of the session that we observed when Aaron explained that Michael was actually their client who was embedded at Pivotal to guide the design, learn how to pair, and build his knowledge of the design system overall.

The two designers were working through a complicated enrollment flow for a new solar energy provider. Clearly, there were lots of details that needed to be worked through, but they understood what problem to solve and were very focused on it. They decided that the process would benefit from adding a step and were working through the interactions in that step. Michael had a prototype up and was pulling together elements from the style guide to create the new step. Aaron was looking at notes and highlighting any missing pieces of information. They took a step back to look at the previous 10 minutes of work. Even though all of the pieces were there, it was clear that users would need a new element to indicate progress in the flow.

Aaron had an idea for an icon, and he grabbed the keyboard to begin illustrating it, pulled in an existing brand element, and quickly cleaned up the layout of the section. They discussed it for a few minutes, identifying a few other approaches they could try, as well. The discussion turned to how the interaction would animate and behave in space, and hands started waving. Then, Michael whipped open Flinto, a simple animation tool, and showed Aaron how it worked. After 20 minutes they had designed the flow, tested it against the product manager’s requirements, and created a simple prototype. Because iteration happens between designers who have the authority to make critical decisions, pairing moves quickly.

The constant switching of roles—driving versus navigating—is a key component of pair design at Pivotal. Although the switching happens regularly, it’s not a battle for the talking stick or one person correcting another. Instead, explains Michael, the switching is about balancing focus on details with longer range thinking. As one person is operating the drawing application or holding the whiteboard marker, the other person is testing the ideas. Does it scale appropriately? Is it consistent with design patterns elsewhere? Does it bring
up issues that the client or product manager will need to think through? How can it be tested?

Another aspect of switching roles is that designers are constantly learning from each other. Maybe one designer is a stronger illustrator and can quickly render a complex visual idea. Maybe another designer has a 3D background and can bring some spatial awareness to the design solution. Or, in the case of Aaron and Michael, maybe they teach each other new tools that are useful to the task at hand.

**Beyond Pairing Two Designers**

Many designers, upon hearing about how design pairing happens, find that even though they might want to practice it, they don’t have enough resources allocated to projects in that way. This struggle for the time and attention to dedicate to design problems is universal, and small design teams are common. So let’s look at how a few other organizations aim to get some of the benefits of pair design without doubling their design teams.

**GreatSchools: The Life of the Designer–Product Manager Pair**

GreatSchools is a national nonprofit that helps America’s K–12 families research schools every year with their website, as shown in Figure 1-3. Michael, a senior product manager, and Dustin, a senior UX designer, didn’t have a lot of experience working together, or in pairs of any kind when they started a project to redesign Community Reviews of schools, which are a key part of the site.
Figure 1-3. GreatSchools helps families research and choose schools
The site began in 1999, and like many sites of that vintage, it had been undergoing a longer-term migration to new core technology. As a result, it was difficult for the team to make assumptions about how the backend systems and processes would or should work. Michael, as the product manager for the new platform, was best poised to help Dustin work through the ambiguity. At the same time, because the system was evolving, there was an opportunity to change systems and processes for the better. However, because reviews are so central to the organization's operations, it needed to match or beat certain metrics of success to be viable. The two decided that if they worked closely together, they could more quickly understand the implications of different designs for both the users, the business, and the system itself.

Dustin and Michael's day-to-day responsibilities were very different, and literally working together, all day, for days was impractical. Instead, they worked in bursts. Dustin ran quick design sessions with Michael workshopping the ideas he'd developed the days before. Michael would review with an eye toward what his stakeholders would want to know, and they then developed surveys or A/B tests to gather the data to back a design decision. Rather than try to “play designer,” Michael says he focused on helping Dustin articulate his assumptions and then develop ways to test them using the site's relatively heavy traffic.

With such a key part of the site up for redesign, they knew that key stakeholders and funders would need to be managed closely and ideas presented clearly, or risk a “kitchen sink” design solution for the minimum viable product (MVP). As they learned what worked and didn't work for users from their research, Dustin says he also began to understand how his work affected marketing, search rankings, and more. Rather than teaching each other design tools or illustration tricks, Dustin was gaining wider knowledge of the core business at hand. Michael in turn, used their research when presenting their work to the stakeholders to justify their design decisions.

Because the problem was complex, and they needed others to fully appreciate the extent of changes they were about to make, they decided to extend their experience of pairing together to working with other teams. So they began a journey of embedding team members from marketing, editorial, and operations, one or two at a time, into their project for a half a day. These sessions were less about co-
creating design and more about taking time to educate different factions about what they had learned and the data they had gathered.

In the end, Dustin says they ended up with a design that maintained a simplicity that was critical to users and met the complex business objectives in record time. By having a product manager as a partner, his design ideas were paired with evidence of successes—from usability findings, to A/B test findings, to engagement metrics that are critical for the organization at large. He found the debates about design elements to be less subjective, and even if it took time to educate other colleagues and funders, that was less time spent defending ideas.

Lab Zero: The Life of the Designer–Developer Pair

Tracey Thompson has been a UX and Visual Designer at Lab Zero, a design and engineering studio in San Francisco, since 2010. Tracey and Ned Holets, a lead software engineer with Lab Zero, have been pairing together as a designer-developer team for the past three years on two large-scale, long-term projects. It began informally—two colleagues in close proximity (their desks adjoin) who began to consult each other on a regular basis as they were working.

They both recall when they realized how pairing together more formally could really change both the process and outcome of their projects. While working on a website to support a PR event, they were facing a tight deadline. They inevitably ended up working some long days, and even some nights to make sure the site was ready for the event. The site design was very high-touch with lots of animation, transitions, and complicated timing issues. Sitting side-by-side, it was easier to refine details quickly, deciding that what was originally specified was quite right. Tracey says she felt like she could better craft the details she knew mattered most to the user interface (UI), having access to a live prototype from which to learn. As some long days grew even longer, they were able to remote pair in the evenings, using a screen-sharing tool, Screen Hero (see Figure 1-4). Although they don't tend to work under pressure like that as much, the partnership they forged continues today on a much different project.
Figure 1-4. Collaboration tools support remote pairing, whether it’s working late from home or in different places altogether

For the past two years, Tracey and Ned have been designing and developing a system for a large commercial bank with many older databases and existing code for a client whose expertise is in many things besides design and technology. Coming off of their successful pairing on the PR website project, they decided to formalize their partnership up front and continue their close collaboration on the new project.

They had mixed results at first. They discovered that each of them needed time and space to do their own research, Tracey into users and what they needed, and Ned into the backend systems and regulations for the system. As Ned put it, “I need to get a full picture of the system, and I don’t want to kill ideas about the design early on by thinking too much about feasibility.” At the same time, Ned felt that coming together around the implications of each of their research bore a lot of fruit. Rather than being on the receiving end of a fully developed idea, he and Tracey spent time working through the highest-level concepts for the system and each of them could
better develop ideas that took advantage of constraints, rather than avoiding them.

Tracey describes one of their first meetings, for which she brought in her research and she began sketching out different concepts and models for the site. She knew that users needed to get a view of their data that was spread across several different services. And Ned knew that calling on all of that data simultaneously was unlikely to be performant, but he also appreciated the user need that Tracey was addressing. Over the course of the design session, they developed an asynchronous approach to the problem that delivered what would satisfy users, in just a slightly different way. This breakthrough turned out to pay dividends over the life of the project because the core concepts and system architecture are designed to scale together.

As they continued through the design of the project, they learned that pairing worked in phases. These days, they collaborate for a few days upfront as they begin the design of any new area or module of the system. Then they break off to do their own design and development work, coming back together again as the sprints turn to the detailed development stages. This flow echoes what happens with designers and other stakeholders who come together and break apart as the ideas diverge and converge.

Ned says this is actually reflective of how pairing works in pair programming. “The myth of pair programming is that you’re doing everything in lock step, but I’ve found that when big breakthroughs happen, engineers break up and work independently on the details.” And Tracey adds that through pairing, she’s become more technical, and there are times when “no one needs to watch me tweak CSS all day.” What’s clear is that by having the ability to work closely together, even across different skill sets, has great value at times. And when it doesn’t pay off, they have learned to maintain communication, while not being joined at the hip.

As this case study shows, designer–developer pairing has many similarities to designer–designer pairing but takes advantage of the more varied skill sets of each partner. By increasing the frequency of communication, Tracey and Ned were able to better understand each other and avoid some of the pitfalls that happen when designers see developers as more distant partners who simply execute the design.
Where Pair Design Lives and Thrives

In light of the case studies previously presented, we can identify some key traits of designers that support successful pairing. We begin to see the kinds of environmental and cultural traits needed within an organization to make it effective.

What Makes a Successful Pair Design Team?

Until this point, we’ve shown how pair design works and how it can be practiced in a variety of ways. And we’ve shown how it can support a variety of skill sets and approaches. For some, pair design can be challenging, and presents an opportunity for designers who struggle to work in real time with a partner to grow some new skills. Across the teams we interviewed, certain key traits emerged as critical to this practice:

Diversity of backgrounds and complementary skills
The individuals within a team are each T shaped, with deep competency in different areas. Although this is not exclusive to pair design, it’s useful to note that pairing two very similar people can backfire as they struggle to hold roles and space for each other.

Quick thinking and strong verbal skills
Being able to talk through ideas in real time is critical to this practice. One criticism that could be made is that those who aren’t native speakers of the dominant language might be at a disadvantage in an environment that is explicitly about efficient verbal communication over more deliberate approaches.

Multistate thinkers who can improvise
The ability to keep a multistate system in mind is critical when evaluating and describing ideas with a partner. Teams often develop a shorthand for communicating complexity as a way to move quickly, but then they must be able to transition into a mode where they can communicate that complexity to outsiders.

Ability to think long- and short-term
A key benefit of pairing is the constant testing of solutions against edge cases and details, without losing track of the bigger vision and project plan. For anyone who will hold the synthe-
sizer role, this skill is critical. Even though all good designers should have this ability, having a dedicated partner helps support this. When practicing solo, it can be difficult to shift your own perspective easily.

**Ability to think systemically and in the instance**

Interaction design is systems work. Most good designers have the ability to keep a complex multistate system in their heads as they work through details. When designers are paired, the team’s capacity to manage the whole system increases. Having one partner explicitly be the “navigator” (versus the “driver” in the Pivotal parlance) helps the team consistently test the system and avoid creating new design patterns unintentionally.

**The Agreements That Underlie Successful Teams**

To operate effectively, pairs must make and stick to agreements about how it is that they should work together. Though this is negotiated idiosyncratically by every pair, there are some ground rules to get started.

**For Those in the Generator Role**

These principles help those fearlessly generating ideas be a good partner.

- **I will show before I tell**
  
  I understand my role is to visualize the conversation in progress. This includes drawings that illustrate the design, but also that illustrate the conversation around it and the decisions that the team has made. I will not erase or delete drawings, so we can review discarded ideas—and the reasons they were discarded—when stakeholders raise them in the future as possibilities. I will accept reminders to visualize when I get too wordy with ideas.

- **I won’t let my ego get in the way**

  I am putting my ideas out into the world expressly so that they can be critiqued. I will not take this personally or defend my ideas just because they are mine. The discussion is not about me or my merits as a designer, it is about the best design.

  Nor will I engage in any “scorekeeping.” In a given session, if five of my ideas are found wanting, it does not mean I am
“owed” a pass on five other ideas. I will acknowledge my feelings of defeat when I feel them and work to move past them.

**For Those in the Synthesizer Role**

These principles help those nurturing ideas to be an empathetic skeptic.

- **I will be specific in my critiques**
  I will acknowledge my “spidey-sense” that tells me when something is wrong about a suggestion and share it with my pair. But if I can’t explain why it is wrong, I will table that feeling so that the design can move forward. I reserve the right to return to the issue if I am able later to articulate a counterargument. I understand that if that counterargument comes too late in the process, we might not have time to make the implied changes in the current project or phase.

- **I will build not block**
  Knowing that ideas are being put into the world to be critiqued, I will work to keep that criticism constructive. I will not simply say, “No, this is not good enough” and expect my gen to produce something new until I am satisfied. I will explicate the problems that I see earnestly and clearly, and work with my pair to identify what is good about the design, and nurture a solution to identified problems.

**For Both**

To do great work as a pair, many of the agreements are about ensuring that both members of the pair feel *psychological safety*; that is, it is safe to take risks.

- **We trust each other**
  It’s a lot of cognitive work to constantly wonder if your partner has ulterior motives, and distrust becomes a self-fulfilling prophecy. For this reason, pairs agree to give each other the benefit of the doubt. We presume good intentions in the absence of evidence.
• **We mutually respect each other**
  Part of what you trust is that your thought partner respects you. Any criticism or defense brought to bear is about the design we are creating together, not an attack against you personally. Even when things get heated, it’s driven by a passion to create the best design for our stakeholders.

• **We will participate in healthy debate**
  It is never about being “right,” it is about what gets us to the best design. This means we are forthright and honest in our conversations, use fair tactics, help each other avoid logical fallacies, are mutually open to being persuaded, and are honest if you are convinced.

• **We will be deliberate about our relationship**
  It's not enough to make these promises at the beginning and then fall back to old habits. Teams promise to take time out of the schedule to look at the relationship, appreciate what is working, and course-correct things that aren’t.

• **Ideas can come from anywhere, but there will be no dueling whiteboards**
  Though the gen is responsible for producing ideas for consideration, the team should take advantage of any good ideas, regardless of where they come from. This includes the synth of course, but also developers, stakeholders, users, and coworkers can all have great ideas for how to solve design problems. When one is presented in the room, though, it will be thought through as thoroughly as we can before introducing another idea from someone else, to avoid a competition of egos.

• **We will defer to tie-breakers**
  To ensure that the project keeps going, we agree to a time limit for being stuck. If after 15 minutes neither of the pair can convince the other on a particular issue, we agree to get another person in the room to whom we can explain the issue neutrally, favoring neither side.

• **We will respect the design process**
  When receiving feedback, we agree not to try and solve problems that are identified, live before the stakeholder, unless explicitly asked to. This is to ensure that problems are consid-
ered carefully and by the team in the same deliberative, acid-tested process pair design is meant to foster.

What Pair Design Needs from an Organization

Some other considerations for designers considering pairing: the teams that we spoke with all recognized that some element of the success of pair had to do with the culture of the organization as a whole. Our interviewees expressed the following as being critical to the success of pair design within an organization:

- The organization must have a commitment to being a learning culture because pair design can take more resources than simply putting a solo designer on an Agile team. Whether because business domains are complex or rapidly changing, some organizations know that they need to foster informal ways to grow, train, and utilize employees. Pair design offers a lot less to organizations that just expect their employees to perform.

- The organization must be quality-focused rather than focused on “shipping” because pair design principally offers major gains in design quality. Across quarters, years, and the lifetimes of products, quality design does yield economic benefits of customer loyalty, brand strength, and less customer support. Crummy design done quick has value to someone, but not to the designer/studio/product line.

- The organization must be customer-focused, and where that focus can drive wins in the marketplace. Of course, this is not to the exclusion of the economic realities that every business has, but an organization that doesn’t deeply value pleasing customers won’t see value in pairing.

- The organization can’t be too vested in its hierarchy or silos. Although there are bound to be levels of expertise among team members, the environment can’t be one in which it’s expected that seniority wins all debates about design decisions. Similarly, pairing might need to occur across organizational units (design and development, for example), and if bureaucratic divisions discourage this, pairs will be unsuccessful.

- The organization must be committed to psychological safety for its teams. (As mentioned earlier in the section on Agreements.) Designers need to feel like they have a space in which they are
empowered to propose half-baked ideas and receive honest critique that is leveled at producing the best design rather than at the designer. If teams don’t have psychological safety from management, ideas will be incremental at best.

Frequently Asked Questions

As we discuss these ideas with others around the world, audience members often ask some similar questions. In this last section, we present three of the most common questions.

My organization/problem is just too large and complicated to work the design with just two people. What to do when you know there will be at least three people in the room?

Recall that pair design is specifically not feedback. Pair designers often must show their work to others intermittently for vetting and feedback. If those “extra” people are not sitting in the room working the problem together, it’s not the same as what we’re talking about.

But if there must be more than one person in the room, as long as everyone sticks to the agreements, has only one person generating at one time, and lets one person playing “lead” synthesizer, it can be managed.

I’m management, how should I implement pair design in my organization?

Our recommendation would not be to simply pull a lever and switch the entire organization to pair design. It would be too much chaos. It’s better to start small. Find the “genniest” designer you can and pair her with the “synthiest,” have them work through a few projects as a pair to see how it goes, evolve a process that works for your organization, smooth out the wrinkles, and become resident experts. Then, split them up, assign them with new pairs, and begin to spread.

I’m a designer, and don’t have the explicit buy-in of my management, but want to try pair design. How should I do it?

There are four recommendations we’ve given out:

Be your own synth

This is a tough mental game, but try to be distinct about your own generative and synthetic modes. Take time to generate, but
then stop, pick up another pen, and then try and seek what’s right and wrong in them yourself. It’s difficult to do, and not ideal, but if you’re a lone designer and there’s no other route, it’s worth a try.

**Find an existing accomplice**

Seek someone in your organization with complementary skills and agree to pair with that individual. Maybe you split your time in two and each generate for your own project while acting as synth for the other’s project. If it succeeds (and we have every reason to believe it will), you can share the success with management to make the case for a wider rollout.

**Get headcount**

If you have the budget and authority to hire another designer, identify which role you’d like to play, and optimize your search for someone with complementary skills. Explain the experiment you’d like to run and try it out with them.

**Go virtual**

There are plenty of designer organizations where you might get feedback: offline through meetups, for example, or online boards such as reddit or ixda.org. You’ll need to scrub your designs or wireframes of any proprietary information, of course, and the virtual nature won’t be as continuous, but might be better than nothing.

---

**How Do I Hire for Pair Design?**

It’s tricky enough to find candidates with the skills and aptitudes I need. How would I also hire for a gen or synth role, when the candidates themselves might not know what that means? I would certainly explain this intended working style to them if it’s going to be part of the job.

After that, seek out *fearless generativity* for gens. Give them a design problem. See if they can work with incomplete information, quickly and clearly visualize solutions, and take constructive feedback to iterate a design in real time. These are the key skills of a gen.

For synths, seek out *empathetic skepticism*. Have a gen design against a problem before them. See if he can ask intelligent questions, articulate what’s good and what’s problematic about a design, and guide the gen in a better direction without picking up the pen himself. Ask
him to write up the design in a paragraph or two at the end, including what next steps might be. These are the key skills of a synth.

In Closing

The practice of user experience design is always evolving, and as we’ve shown in this report, pair design is one contribution that can yield a lot of benefits for teams. If you are looking to organize a team of designers for your organization, or if you’re a designer who’s feeling a bit out on a limb all alone, we hope you’ll find practical things to try.
About the Authors

Gretchen Anderson spent the first part of her career in design consulting for firms like frog, Cooper, LUNAR, and Punchcut. Recently, she served as the VP of Product for GreatSchools. Currently, she consults with several clients including Auris Surgical Robotics on the design of the hardware and software of a next-generation surgical system. She has always tackled complex design problems and finds working in a close pair to be a big part of success in taming that complexity.

Christopher Noessel is a veteran of more than 25 years in the interaction design industry, having owned his own small studio in Texas, working with the Futures Prototyping group at Microsoft, and working at Cooper as an interaction design consultant for a decade. In that capacity he helped develop, teach, and speak about pair design around the world. He also managed the practice of gens and managed teams working this way. He is now the global design practice manager for the travel and transportation industry at IBM.

Thank Yous

Christopher would like to thank the many synthesizers with whom he worked at Cooper for deliberately discussing and evolving practice with him, but especially Suzy Thompson, with whom he worked for many years together and developed some of these early ideas into a presentation at South by Southwest.