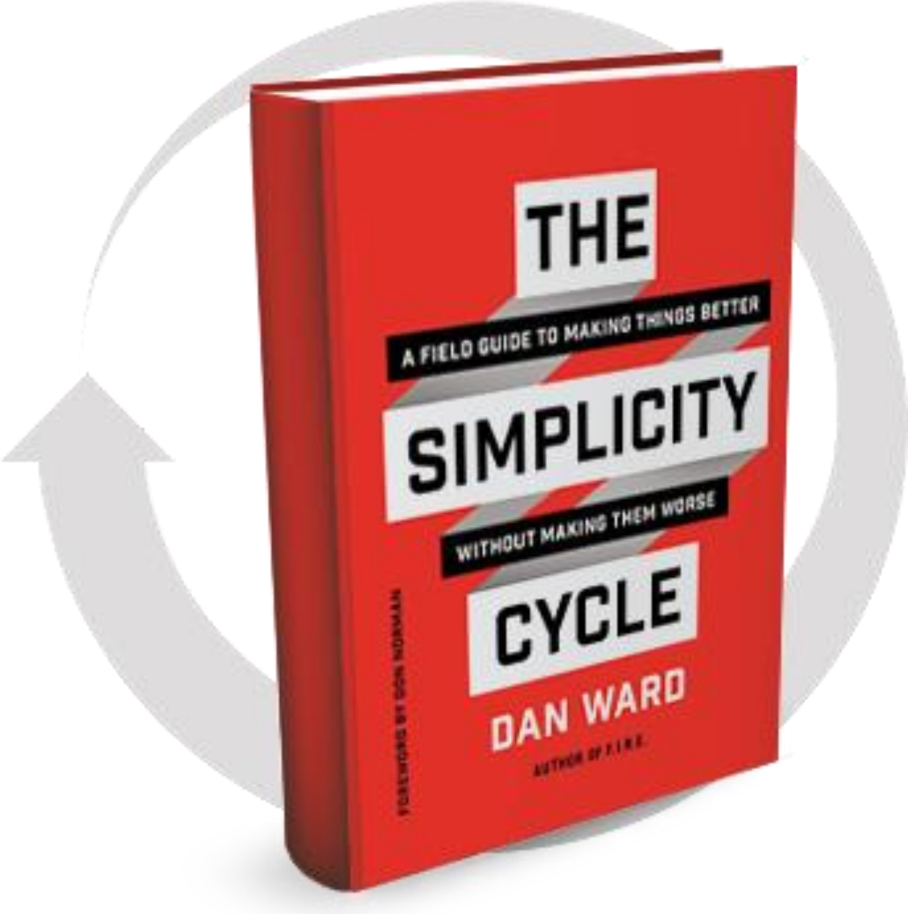
The Simplicity Cycle

Strategy Workbook



INTRODUCTION

This workbook walks you through the process of mapping out a project development strategy using the Simplicity Cycle framework.

The Simplicity Cycle is applicable to a broad range of activities, from writing code to designing business processes to creating a PowerPoint presentation. How will you use it? That's up to you, so take a moment to decide on a project or product to use as we go through this workbook.

To get the most out of this workbook, it helps to be familiar with the Cycle diagram. If you're not already acquainted, there are several ways to get up to speed:

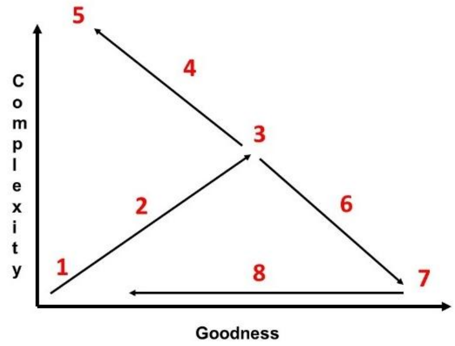
> **Read *The Simplicity Cycle* book**. It's the most complete treatment of the idea, and you can download the first chapter for free at my website.
>
> In a hurry? **Get the free Simplicity Cycle Manifesto** at ChangeThis.com. It's a snack-sized version of the concept and is sufficient to get things started.
>
> *Really* in a hurry? Take a look at the 1-page **Simplicity Cycle Quick Reference Sheet** at the end of this workbook.
>
> Prefer watching to reading? Check out this video of my **Simplicity Cycle presentation**.

Once you have a basic familiarity with the Simplicity Cycle, you'll be ready to use this workbook to build your strategy.

THE EIGHT STEPS

We can break down the Simplicity Cycle into the eight phases or steps shown below:

1. **START**
2. **The Complexity Slope**
3. **SHIFT**
4. **The Complications Slope**
5. **STOP**
6. **The Simplification Slope**
7. **SHIP**
8. **The Time Arrow**



At each step, we'll use a different set of techniques, so knowing what step we're in and which techniques to use is the key to moving through the Cycle efficiently.

It is also important to plan ahead and keep the future steps in mind, because the decisions we make at the beginning of the project will have implications down the road. A little strategic thinking early on can save a lot of time and frustration later.

For example, knowing we're going to eventually delete some components improves our design decisions in three ways:

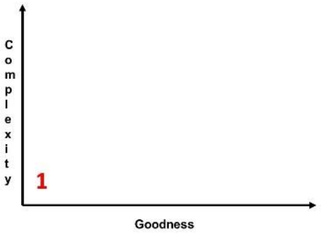
> First, it makes it **easier to add something** even if we are not 100% sure the addition is a good idea because we know we can always subtract it later.
>
> Second, it encourages us to adopt a **modular approach** to the architecture, making it easier to move, remove or replace components in future iterations.

> Third, it makes it **easier to subtract something** later in the project, because the idea of taking things away is part of the plan from the start, so we get less attached to any particular piece and are unsurprised when a deletion is necessary.

This workbook will guide you through the process of planning out your strategy for each step, before you actually design, add, or delete anything. It should smooth out the way ahead and help you avoid painting yourself into a corner or encountering any unfortunate surprises.

To help you track where you are and where you’re heading, I’ve included a mini-map at the beginning of each section. It identifies the specific step being discussed in each section.

Ready to go? Let’s get started!



STEP 1: THE START POINT

"*One of the hardest things for a beginner to do is merely to get started*."

- Charles Schultz

In *The Simplicity Cycle Manifesto*, I explain that when we are faced with a blank page or empty canvas in the early phase of a project, there is just one design move available: **add something**.

Upon further reflection, I realize that is not exactly true. There is another option. We can choose to do nothing. Instead of starting, we can delay and hesitate.

This is not a good option, but it is all too common. It is easy to busy ourselves organizing the paperclips, launching another study, changing "happy" to "glad" and back again, or looking for another data point to guide our action… someday.

We have a hundred different ways to stay occupied without actually beginning the work. This is a mistake, and it's one you don't want to make. As Hugh MacLeod explains in one of his brilliant Gaping Void cartoons,

"*You don't need more time. You need to decide.*"

The first decision we need to make is the decision to start. It happens here, in Step 1.

Begin by identifying some project specifics and making a list of things you plan to add. One way to do this is to create an outline or wireframe diagram that identifies the various parts, components, and pieces you're planning to use.

Writing a report? Your list might look like this:

Introduction
Conclusion
Three Case Studies
One Research Finding
Three Tables
Six Diagrams
Seven Ideas

Creating a website? Perhaps your list will include:

Landing page
Bio page
Content page
Shopping cart

The list doesn't have to be in the right order at this point. We are just collecting some components at this point, and we will organize it all later. The list also doesn't have to be complete. That comes later too, when we realize our report needs a bibliography or our website needs a contact page.

As we get started, we should expect our list to include things that won't make it to the final cut (Six diagrams in our report? Maybe four or five will suffice). For now, the goal is just to start brainstorming the pieces, parts, and components that will go in to our final product.

The point is to get **STARTED**.

Grab a writing instrument of your choice, turn the page, and start listing the things you're going to add to your project. Feel free to draw a picture if numbered lists aren't your thing.

My project is:______________________

The customer is:_____________________

Things To Add

1)

2)

3)

4)

5)

6)

7)

8)

9)

10)

11)

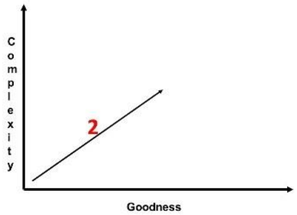
12)

It's not too early to ask **why** we're adding all this stuff. What value are we seeking to provide to the customer, what are the measures of merit, the desirable attributes of our project? In other words, what sort of "goodness" are we trying to create? In the space below, write some words to address those questions:

What are the specific measures of merit for this project? What are the top three or four requirements we're trying to satisfy? Take a moment to write down some words that address these questions. As with our list of things to add, these words don't have to be perfectly precise or accurate. They just have to be written down so we have something to work with.

1.
2.
3.
4.

**Great work! Now that you've got things started,
it's time to move on to Step 2.**



STEP 2: THE COMPLEXITY SLOPE

Once we’ve sketched out the skeleton, outline, or wireframe, Step 2 involves actually filling things in and looking for gaps. This is where we begin to write the content, assemble the code, or make the first few PowerPoint charts.

This is also where we discover we overlooked an important part or three. When that happens (and it is *When,* not *If*), we add it in. That’s all part of the process.

The trajectory we’re on involves making things more complex and better. It’s a messy process, but it’s a productive kind of mess.

It’s important to let it be messy. I know this can be difficult for some of us. Resist the urge to sort and filter and arrange things at this point. Don’t let yourself be distracted by organizing or cleaning up yet. Don’t edit, rephrase, or delete anything. Keep it all.

> **Our focus should be on creating things, identifying the missing parts, and adding to the pile.**

In this early stages of a design, the most effective tools and design moves are *additive*. Now is not the time to take things away or polish them up, because there’s a good chance we’ll end up polishing and refining something that won’t make the final cut or we’ll remove something that we’ll need to add back in later.

It’s simply too soon to do that part. Don’t worry, we’ll get there. We’ll make things neat and tidy… just not yet.

So give yourself permission to add with wild abandon. This will make the next steps much easier and more productive.

As we focus on adding things, there are several techniques we can use. Here are eight to consider:

1. **Introduce**: Insert a previously absent piece into the design.

2. **Replace:** Exchange an existing element with an alternative piece. This strategy is additive in the sense that it introduces a new component, but it also involves removing something from the design.

3. **Divide:** Split a single element into multiple elements. Rather than one superhero with 10 powers, create a team of superheroes with one or two powers each.

4. **Expand:** Transform an existing element into a larger version of itself.

5. **Add copy:** Introduce redundancy. These could take the form of a hot-backup that takes over a function when the primary system fails, or a copy that performs its function alongside the original, as in the dual wheels on a tractor trailer.

6. **Monitor:** Insert a check function to confirm an existing element performs as designed. A monitoring function generally produces a signal to indicate status and may automatically trigger a switch-over to a back-up system (see #5 above).

7. **Roughen:** Increase friction between design elements, either in the form of a time delay or a more physical degree of friction (texture, layers, etc). The right amount of friction can improve traction and prevent slippage.

8. **Accelerate:** Introduce an element sooner in the process than planned. This is a form of time-shifted addition, where the element was always intended to be part of the design but is now included earlier.

The point is to assemble a big collection of parts. Make some parts bigger. Chop others into smaller bits, but have lots of them. Make some redundant copies. Add a little friction here and there. Many additions will be removed later, so give yourself permission to add weird, funny, unusual, non-standard addition into the mix.

Now that you've seen some tools, it's time to create your strategy. Take a little time to write down some words in response to the following questions:

What sort of additions will you do?
(*Words, paragraphs, chapters, sections; Functions, interfaces, databases; Charts, diagrams, pictures, tables, calculations)*

How will you add to your design? What would it mean to use each of these tools on your project:

1. Introduce
2. Replace
3. Divide
4. Expand
5. Add Copy
6. Monitor
7. Roughen
8. Accelerate

Knowing that many of your additions will be removed later, how can you make your additions severable and modular in the first place, so that these removals will be clean?

What weird, funny, unusual, non-standard things will you add to the mix?

Let’s also revisit the top three requirements from the previous section. List them here, and give a check-mark to each one that is satisfied or supported by the planned additions in the list above. Are the top three requirements each addressed by the initial additions? Are there other requirements we previously overlooked? Write those down too.

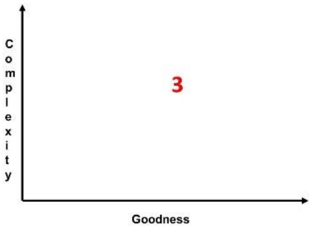
□

□

□

□

Awesome! Now we’ll head to the Shift Point,
where things really start to change.



STEP 3: THE SHIFT POINT

When our design hits the critical mass of complexity, it’s time to shift our behavior. We’ve come to the point where we stop adding things and adopt a different set of tools, techniques, and methods.

It is incredibly easy to blow right past this point, continuing to add long after such additions ceased to be productive. That’s particularly common when we’re not paying attention or if we don’t know what to look for. That’s because the process of getting to this point produced additive inertia in our project. We’ve developed habits and built up some momentum that we need to keep an eye on.

To make sure we’re paying the right kind of attention to the right things, it’s helpful to keep an eye on a handful of signs we can easily monitor throughout the process and establish automatic triggers to nudge us in a more productive direction. These built-in checkpoints remind us to periodically stop and assess our progress, checking to see whether we’ve hit the critical mass of complexity and need to shift from additive to subtractive mode.

Depending on the nature of the project, we might use any or all of the following techniques:

> **Timeboxing**: Shift after a pre-determined amount of time spent adding. This is most effective when you have a hard delivery deadline.
>
> **Size/Weight**: Shift after the design has reached a pre-determined size, weight, parts-count, planned functions. This is most effective when the product has to meet certain size or weight requirements.
>
> **Feedback**: Ask an objective third party, a colleague, a mentor, or customer for their opinion on whether we’ve got enough. This is almost always a good idea.

Comparison: This probably isn't your first rodeo, so compare this project with an earlier one. If this *is* your first time, no problem. Just find a similar project someone else did and use it for comparison. What components did that project include? Does your project have all those pieces?

Intuition: We all have a gut feel for things, a phenomenon which is often undervalued. I don't recommend making a habit of basing decisions *solely* on intuition, but I also don't recommend ignoring the message your gut is sending. It's often pointing to something important, correct, and reliable.

Rubric: One of the most obvious ways to tell whether you've checked all the boxes is to look at the original statement of need, the expressed goal, or the requirements list. Most projects have some sort of rubric that defines the standard to be met, so take a look and see if you've covered all the bases.

The question to ask at this checkpoint is not "How does this look?" but rather "Did I miss anything?"

That's an important distinction. We're not trying to make it pretty. At this point we're going for "complete."

The objective is to identify whether any essential functions, components, or messages have been omitted. We've cast the net wide and pulled together a big pile of potential pieces already. We won't keep them all, but we don't want to overlook any either. If some parts in your list or stack seem extraneous, unconnected, or unnecessary, that means you're doing it right.

Time develop your shift plan. Fill in the blanks on the next page.

My Shift Plan

I'll take a pause to assess whether to shift from additive approaches to reductive techniques:

After _______ days / weeks / months

After producing _________ components (slides, functions, words, steps)

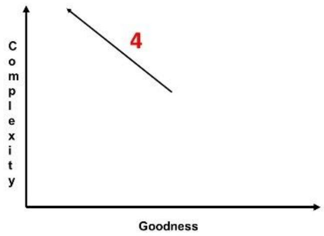
After satisfying __________ percent of the items on the rubric (70%? 120%?)

Based on feedback from _______________, which I will solicit ____________ (daily, weekly, after every 4 / 6 / 10 additions, when the design is ________ big)

EXAMPLE: You are working on a presentation and decide to pause and assess after three days (*timeboxing*). You followed the prescribed format (*rubric*) but nevertheless the presentation looks incomplete to you (*intuition*). Just to be safe you talk with a customer, mentor, or colleague (*feedback*). They suggest the presentation should also address a few additional topics, so you spend three more days adding those in and re-evaluating the objective of the presentation (Convince? Inform? Entertain? All of the above?) before seeking additional feedback.

Following the example above, write a few words or sentences describing your trigger & checkpoint plan:

Great job! These triggers and checkpoints will help you keep you on track!



STEP 4: THE COMPLICATION SLOPE

I call this Step 4, but it doesn't have to be. We're actually better off skipping straight ahead to Step 6. If the triggers and checkpoints from the last section work the way they are supposed to, that's exactly what will happen.

Unfortunately, enough of us (including myself!) find ourselves here on a regular basis, so it's worth taking a few moments to understand what's going on and make a plan for dealing with it.

Our strategy at this point has two components:

1. Recognizing we're headed in the wrong direction.
2. Doing something about it.

The signs of excessive complexity we'll watch for along this slope are similar but somewhat different than the signs we looked for in the previous step. Instead of indications that we've hit the inflection point and are ready to begin subtracting, we're now keeping an eye out for signs that we have gone a mile past that point.

It's a difference in degree that becomes a difference in kind.

Where does this excessive complexity come from? It could be designers, writers, or engineers who only use additive techniques and never subtract anything. It could be customers, bosses, or other stakeholders who keep asking for *just one more thing*. It could be a policy, regulation, or other guiding principle that drives our project to address certain requirements or contain certain elements. Or maybe it's just a prevailing ethos that says "more is better."

Here’s a list of things to watch for:

An excessively excessive amount of redundant redundancy which is excessively redundant, to an excessive degree. Also, redundant.

The size, weight, or timebox threshold is significantly exceeded.

You are bored, confused, or both.

Debugging or editing is painful, slow, and unproductive.

Test results are unpredictable (because we don’t understand how the thing works).

Things feel unfocused and misaligned.

Project objectives are difficult to explain; external observers can’t tell what the project is supposed to do.

The list of requirements has expanded significantly.

Changes to the design have a minimal impact on performance.

Changes to the design have a negative impact on performance.

Recent versions of project provide less value than previous versions.

What can we do in this situation? Well, there are a few options:

Call a halt

Freeze the baseline

Take a pause to disrupt the momentum

Reexamine the definition of Goodness

Get a blank sheet of paper and start over

The time to develop a response plan is *before this all happens*. The plan should include a watch list and a few pre-arranged courses of action, along the lines of "When we see X happen, we will do Y." Take a few moments to make your plan in the following pages.

Reversing course on the Complication Slope

Potential signs of excessive complexity on this project are:

Size / weight / cost exceeds _______________ (quantity or percentage)

Schedule slips by _________________

If _______ consecutive additions do not convey increased value

If _______ consecutive tests provide unexpectedly negative results

If _______ people ask "What is this supposed to do again?" in a given week

Likely locations, sources, and causes of excessive complexity on this project:

1.

2.

3.

Action to take

Revisit the definition of "Good."

Identify sources of unnecessary complexity (i.e. which components are increasing complexity rather than adding value?).

Shift from additive to subtractive mode.

Set aside the tools from Step 2 and decide to move to Step 6.

And as always, we need to keep our eyes fully on the prize, making sure the project’s trajectory is heading in the direction of increased goodness. So jot down a few notes about the measures of merit and signs of value for this project.

The measures of merit for this project are:

1.

2.

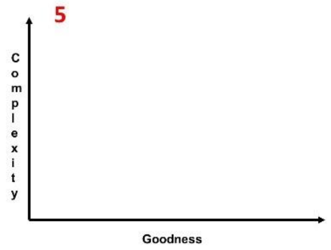
3.

The customer’s definition of goodness includes:

1.

2.

3.



STEP 5: THE STOP POINT

Hey, it happens to the best of us. We get excited, carried away, or distracted and all of a sudden things end up way more complicated than anyone ever wanted them to be.

Maybe our early additions weren't as helpful as we thought they'd be, so we continued to add in the hopes that the next piece would add more value than all the previous pieces.

Or maybe the initial additions were awesome, so we kept doing more of the same in the hopes that the improvements would continue indefinitely.

Surprise – they didn't!

Our triggers and checkpoints didn't work, our strategy execution wasn't effective, and we end up staring at the equivalent of the US Tax Code. That's not good. That's not good at all.

This calls for drastic action. It's not enough to shift our behavior (although yes, we'll need to do that). We need stronger medicine, the equivalent of a tourniquet rather than a Band-Aid. It's time to implement our *Excessive Complexity Emergency Response Plan*. Fortunately, the plan is pretty simple:

> **Step 1**: Admit things are bad. We got addicted to complexity and things went sideways.
>
> **Step 2**: Stop adding things. Stop doing anything at all. Put down the pencil and go for a walk. I'm not kidding.
>
> **Step 3**: Grab an eraser, scissors, machete, or match and go to work. Or just take out a blank sheet of paper and start over.

The most effective strategy to extract ourselves from this mess is to recognize in advance that we might get into it. Admit it's a possibility. This way we can say “Hey everybody, we knew this might happen. It’s not a surprise. We have a plan. Let’s get to work.”

This is a crucial step, even if the “everybody” is just you.

The alternative is to be shocked (*SHOCKED, I tell you!*) by the fact that complexity ran amuck, to be completely gob smacked that it happened to us. Who knew complexity could grow out of control? Wow, I did not see that coming!

The good news is we’re smarter than that. The possibility of ending up in this situation despite our best efforts won’t surprise us. We know it could happen, and that’s the secret to dealing with it. By admitting it may occur, we’re more likely to be ready for it when it does occur.

Just like in Step 3, we’re going to create some triggers that will point to some very specific actions designed to get us back on track.

This plan works best if people know about it, so we’re going to make a list of people who need to be included. Depending on the project you’re working on, this might include coders and engineers, writers and marketing folks, customers and collaborators, executives and managers. Pretty much anyone who is involved in adding things to the project and who thus will need to be part of the Stop activity, alongside anyone who may feel nervous about the idea of halting activity, however important, effective, and temporary this halt might be.

Excessive Complexity Emergency Response Plan

The person with the authority to activate this plan is:______________

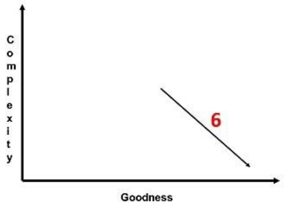
People who need to be informed in advance that the *Excessive Complexity Emergency Response Plan* exists include:

1.
2.
3.
4.
5.
6.

The activities that will be stopped include:

1.
2.
3.
4.
5.
6.

OK, you've got the Stop plan in place. Bravo!
Now take a deep breath and get ready to start again.



STEP 6: THE SIMPLIFICATION SLOPE

"Only the hand that erases can write the true thing."

- Meister Eckhart

In the early phase of a project, our most effective tools are additive. Now we've come to the point where we set those tools aside and pick up an entirely different set. We begin to use subtractive tools and reductive thinking modes. We're talking about "creation by subtraction," more like sculpting than painting or sketching.

Whether we got here straight from Step 3 or took the longer route through Steps 4 and 5, this can be a tough transition to make. Why? Simply because the techniques and activities are so different than what we've been doing up to this point.

The good news is these subtractive tools are not intrinsically harder to use than the additive tools. In fact, I think they are *easier* to use. But… they're new and tend to feel unfamiliar when we first start using them. So be prepared to feel a bit awkward and wobbly at the beginning of this step. Take it slow at first, try a few small cuts here and there, and understand that with any tool, mastery takes time.

Once you get the hang of it, it's actually quite fun. Not sure how to start? As we shift our focus to subtracting things, there are several techniques we can use. Here are eight to consider.

1. **Delete.** Remove an existing piece from the design. This deletion may be a piece that is redundant, extraneous or obviously unnecessary. In other situations, we may remove a piece that appears to be useful or even essential, and we just might discover that the design performs better without it.

2. **Trim**. Remove a portion of an existing element. This strategy is subtractive in the sense that while the component itself stays in the design, it no longer contains all its original attributes or functions.

3. **Integrate.** Combine multiple elements into a single element. Rather than two or more distinct parts, they are synthesized into a unified component.

4. **Shrink.** Transform an existing element into a smaller version.

5. **Remove copy.** Reduce redundancy. This strategy is useful if a design element is highly dependable and has an un-used backup, if the system's performance is generally unaffected by the failure of a particular component which thus does not need a built-in backup, or if a replacement can be quickly introduced when a component fails.

6. **Trust.** Remove a piece that provides a check function and trust that the remaining components will perform as designed (see No. 5).

7. **Polish.** Reduces friction between components.

8. **Delay.** Introduce an element later in the process than planned. This is a form of time-shifted subtraction, where the element is always intended to be part of the design but is now included later than originally planned.

In Step 3 we asked if the top three requirements were each addressed, to make sure we didn't leave anything out. Now the question is reversed. We want to make sure every addition is aligned with a top priority requirement.

The objective is to identify any pieces, parts, or functions that aren't linked to a top requirement. These are the prime candidates for removal.

Easier said than done, I know. Nobody wants to remove their straw from the camel's back. The key to successfully deleting things is to focus not on deletions but on goodness. Emphasize the final objective, the important requirements, and look at the ways each reduction increases value.

The question to ask is not whether a thing is a good idea but whether we can do without it. **The guiding principle is that if we can remove it, we should remove it.**

It's a mind shift to be sure, and as with every step in this process, it is much easier to execute if we plan for it in the first place. Establish a common understanding among all involved parties that many of the initial additions will not make the final cut. The fact that we're going to subtract things from the project shouldn’t surprise anyone. Telegraphing our intent to delete helps prevent undue attachment in the first place.

Remember in Step 2 when I said to resist the urge to organize and polish things? Now you get to indulge that desire as much as you want. This step is where we sort, sift, filter, and bin things, considering each piece and making sure it has a home in the overall project. Thing that don’t have a home, pieces that don’t contribute value – out they go!

This is where the project’s structure really begins to emerge, where the sense-making happens. Now we’re making judgments about what belongs and what doesn’t, what helps and what hinders.

This is a collaborative activity, so don’t just fill this next section out. Talk about it with the other stakeholders and contributors. Let them in on the plan and discuss ways to make things better by making them simpler.

This approach to simplification works best when we do it *with* our partners, not *to* them.

The Simplification Plan
People who will be involved in / informed of these subtractions include:

1.
2.
3.
4.
5.
6.

What sort of subtractions will we do?

(*Words, paragraphs, chapters, sections; Functions, interfaces, databases; Charts, diagrams, pictures, tables, calculations)*

How will we subtract from the design? Think about what it would mean to use each of these tools on the project:

- Delete
- Trim
- Integrate
- Shrink

- Remove Copy

- Trust

- Polish

- Delay

Let’s once again revisit the top three requirements from the previous section. List them here, and give a check-mark to each one that is truly essential. We may find that one or two can be removed from the list.

□

□

□

Fantastic work – we’re almost there!

Complexity
7
Goodness

STEP 7: THE SHIP POINT

Congratulations! We've now arrived at a point where complexity is very low and goodness is very high. This was the goal, and we've hit the target.

At this point, the project is streamlined and elegant. We've assembled all the components needed to achieve the most important functions, and removed all the unnecessary components. The final product now does everything it needs to do and nothing it doesn't need to do. It satisfies the rubric completely, and is not bigger, heavier, or more expensive than we want it to be.

Take a moment to bask in the glory of this achievement.

There is now just one thing left to do: ship it.

Like so many steps in this process, that is easier said than done. There's always a temptation at this point to continue touching the frosting even after it's smooth (don't do it!), to give the thing one more touch of polish, to make it a bit more perfect.

These little tweaks and changes don't make things better or simpler. They just delay the ship date, increasing our exposure to change (more about that in step 8). If we wait too long, the moment may have passed.

To avoid making this mistake, go back to the measures of merit, the definitions of goodness, the stated value we're aiming to provide. Revisit some of the triggers we discussed in earlier steps. Recall, these may be based on time, size, weight, or functionality.

Whatever triggers or signs we decide to use, it's important to make sure the whole team understands what done looks like. This way we can head-off any late-breaking reluctance to ship. When we hit the trigger, we can ship because we've already agreed to do so.

The Ship Strategy

Indications the product is ready to ship include:

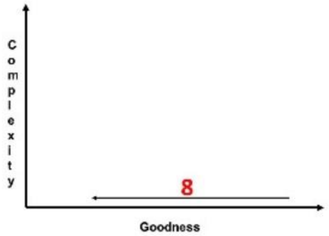
1.
2.
3.

Who will confirm it is ready to ship:

1.
2.
3.

How will I ship it:

1.
2.
3.



STEP 8: THE TIME ARROW

As much as we might like to rest on our laurels, the universe has other plans. And no matter how awesome and simple our project might be, time has a way of making it less awesome.

The reason is simple: change happens. Economies change, technologies change, people change. Your customers have new options, new interests, new expectations, and new priorities. Products that were a great fit yesterday eventually become ill-suited to tomorrow's market. What was once elegantly simple now appears inadequate, dated, or incomplete.

In other words, time pushes goodness to the left.

Good news: we know this will happen. It's not a surprise. So we get to respond to those changes. That generally involves a return to Step 1, where we once again pick up those additive tools.

As with most of the steps in this cycle, this is much easier to deal with if we see it coming. It's not too soon to plan for our project's eventual obsolescence. We may not be able to predict exactly what changes the future will hold, but the fact that the future will be surprising should not itself be a surprise.

Take a few moments to think about the types of changes that might come your way. Some will be easy to predict, others we may never see coming. That's ok. We're going to make a plan to deal with both categories of change.

We don't need to spend a ton of time on this or get hugely specific. The key is to remember that there are no facts about the future, so plan accordingly.

The Time Arrow Plan

Economic changes could affect the project's goodness in the following ways:

1.
2.
3.

Technological changes could affect the project's goodness in the following ways:

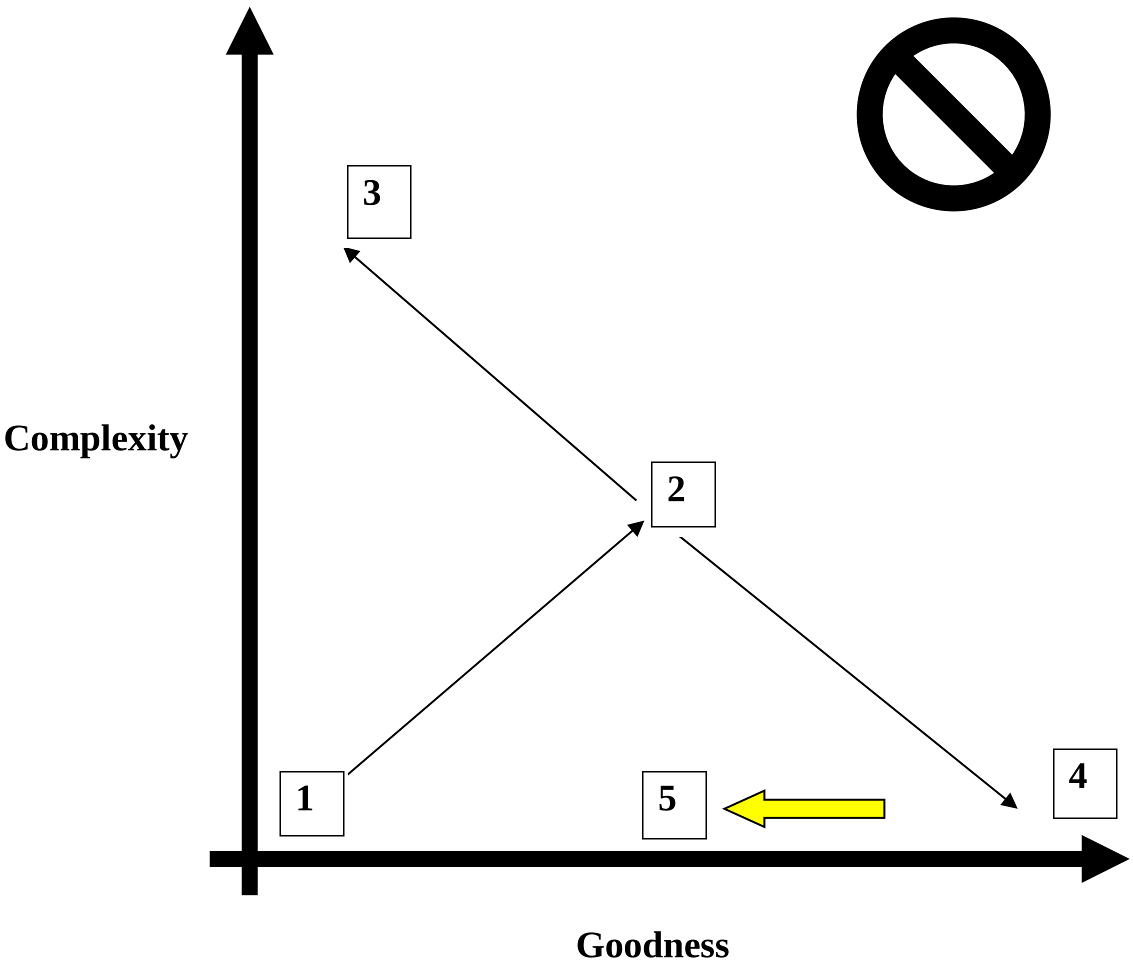
1.
2.
3.

____________ changes could affect the project's goodness in the following ways:

1.
2.
3.

Every ending is a beginning, so print out a fresh copy of this workbook and start thinking about what comes next!

The Simplicity Cycle Quick Reference



1 – The START point. The design is simple, basic, immature, and delivers *little value*. The best move involves additive strategies that increase complexity.
2 – The SHIFT point. The design has accumulated a critical mass of complexity and now delivers *significant value*. Now the best move is to adopt reductive strategies that decrease complexity.
3 – The STOP point. The design has accumulated too much complexity, which *overwhelms value*. The best move is to take a pause, then significantly reduce complexity.
4 – The SHIP point. The design is elegant: simple and effective, providing *maximum value*. Declare it complete and send it out into the world!
5 – The yellow arrow indicates time pushing things in the direction of decreased goodness, as yesterday's breakthroughs become tomorrow's commodities.
This brings us back to point 1, where the cycle starts all over again.

A NOTE FROM DAN

Thanks for picking up this copy of *The Simplicity Cycle Strategy Workbook*. I hope you found it helpful, and I'd love to hear what you think of it. Drop me a note on Twitter (@thedanward), on LinkedIn (https://www.linkedin.com/in/thedanward), or via the contact page on my website (http://www.thedanward.com/contact/)

To learn more about innovation, program management, or simplicity, stop by TheDanWard.com. While you're there, you can download lots of free resources and sign up for my Quick & Simple Newsletter.

May all your projects be elegant,
Dan