Agile Business Analysis

Think of the software or the website you are about to build, the consumer product you want to bring to market, the service that your organization is planning to provide, or the device that you are building to sell. Whatever it is that you're working on, let's call it your *solution*. What's the most important quality your solution must have? Or, to put it in a different way, what is the single quality that, if missed, would cause the solution to be a failure?

Of course your answer is *meeting the needs and solving the problem for the people who buy it or use it*. It does not matter how quickly you build the solution, whether you come in on time and on budget, if you use an agile approach or traditional methods, or if you build it yourself, outsource it, or assemble it from components. If you don't solve the real problem of your customers or users, your solution is worthless. Actually, it's less than worthless. You've wasted your time and resources building it.

> ### agile.
>
> When we use the word *agile* in this book, it means small "a" agile. Big "A" Agile refers to one of the Agile methods. We are agnostic as to which Agile development method you are using. We use *agile* to discuss the ability of the business analyst to respond to changing circumstances and unexpected discoveries and to deliver value.

This is a book about agility in business analysis and how it helps you discover your customer's real problem, find the real needs, and deliver better, more valuable, more relevant solutions.

Why Is This "agile"?

We are using the word *agile* in the sense of being nimble, active, lively, flexible, adaptable, or quick. We do not use the word with a big A. Big "A" Agile means that you are following one of the Agile methods—Scrum, XP, Crystal Clear, Lean, Kanban, DSDM, or any of the others. We are not going to talk about these—they are plentifully covered elsewhere—but everything you find in this book is compatible and applicable to any Agile method.

This is not about better ways to build solutions, but ways to build better solutions.

This book is about doing business analysis in an agile way—trying new things, adapting to changes, adapting to things you discover, being flexible in your approach according to your circumstances, and being quick. Experience tells us that duration matters and that the quickest way to deliver a solution is to deliver the right solution. Indeed, our projects should be as short a duration as we can possibly make them. Prioritizing relentlessly,

investing only in solutions that deliver value, and ensuring that our learning is always fruitful leads to the quickest and happiest results.

Möbius Strip

Figure 1.1 shows a *Möbius strip* representing the continuous nature of constant delivery. If you are not familiar, the Möbius strip—named after German mathematician August Ferdinand Möbius—is made by taking a strip of paper, making one half twist, and joining the ends to make a loop. The result is a two-dimensional entity with only one surface. Start anywhere and draw a line along the surface; your line will arrive back, uninterrupted, at the point you started.
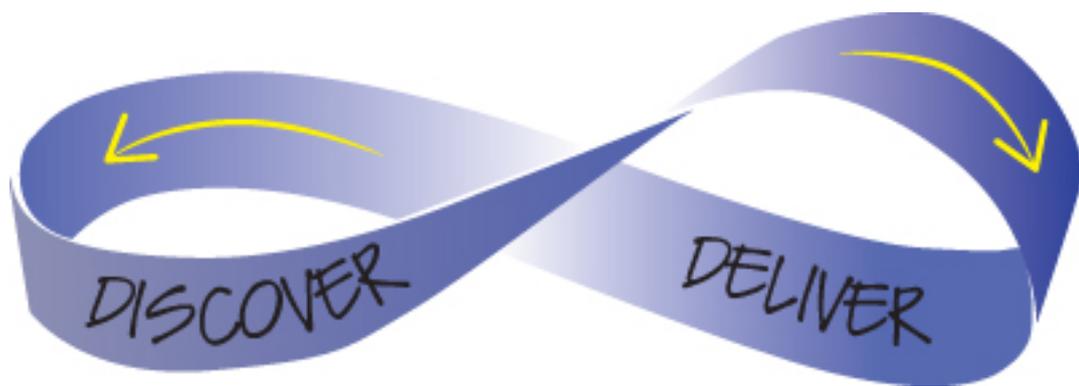
Figure 1.1

Agile development has two aspects, but it is really one continuous process, so we can represent this situation as a Möbius strip. One part of this strip is the Discovery of the problem and its needs, and the other is Delivering a solution that implements those needs. This book is mainly about the Discovery side of things. This diagram is adapted from Ellen Gottesdiener and Mary Gorman's graphic in their book Discover to Deliver.

We see this uninterrupted characteristic of the Möbius strip as an appropriate metaphor of the continuous process of Discovery and Delivery.

Discovery and Delivery are a continuous loop.

The Discovery side of the loop is concerned with discovering the customers' needs. This means uncovering the customers, their objectives and values. It involves experimenting with proposals as a way of exposing the real problem.

Once the optimal solution to the problem has been discovered, it must be communicated to the Delivery portion of the loop. This communication is done in a way that enables the Delivery team to deliver the correct product. It is also done in a timely manner that allows the Discovery and the Delivery to work synchronously. This in turn makes feedback from delivery more useful to the discoverers.

Discovery and Delivery are a continuous loop; the loop is continuous until either all the customer's needs have been satisfied or it is decided that anything remaining is of insufficient value.

But it doesn't end there; the looping continues after a suitable product has been delivered.

The world we live in is agile; it is constantly changing and then reacting to its own changes. There are always new opportunities, new problems, new technologies, and new

innovations that lead to new possibilities; new laws that mean changes to our businesses and solutions; and changes that mean more Discovery and more Delivery.

Why Are We Concerned with Business Analysis?

At the beginning of many development efforts, people *assume they know the solution*. Given the number of failures in product, service, and software development, they clearly don't. Delivering a solution is not the same thing as delivering value. Sometimes it's the opposite.

Any sensible approach to solution development must involve taking whatever time is needed to work with our customers to discover what is most valuable to them, what their real needs are, and what the best solution is to satisfy those values and needs. We should also ensure that the customer's problem is worth solving and that the solution will bring value to both the customer and the provider of the solution.

Delivering a solution is not necessarily the same thing as delivering value.

One of the foundation ideas of agile development was to avoid "big upfront activities." It implied that teams should not spend too much time building a complete requirements specification before beginning development. Unfortunately, some overenthusiastic acolytes took this to mean "nothing upfront." But "nothing upfront" just doesn't work; anything meaningful has *something* upfront. If you want to drive to Baltimore, you must first find out how to get there or program your GPS. This is an upfront activity and is far preferable to setting out to drive along random roads hoping to find one that leads to Baltimore. Acting randomly is not going to get you where you want to go. You would not attempt to provide streamed music without first understanding the audience for your music and how they might use it. Nobody attempts to build a house without planning how the rooms are laid out and where the doors and windows should be.

Software projects cost the same as, or more than houses do. Why should anyone think software can be built without *any* upfront preparation?

Software is certainly more malleable than bricks and mortar. But if you are not solving the right problem, no amount of refactoring is going to save you. If you fail to understand the customers for your solution, no amount of redesign will build a product they want.

However, despite the need for some upfront activity, it must not take too long. Nor is there a need for the upfront activity to be finished before construction work begins, to produce voluminous documentation, or for the upfront activity to be remote and unconnected to the building activity.

Despite the need for some upfront activity, it must not take too long.

This book is about integrating analytical thinking into any development process. It aims to show you how to ensure that the right solution is delivered. It also aims to show you that by not heedlessly constructing an assumed solution but by taking the time the solution needs to discover the real problem, you take giant steps toward the success of your mission.

Let's look at an example to see how this works. This might seem at times a little long-winded. Please keep in mind that the activities discussed are overlapping and iterative. They can be done quite quickly, and in Chapter 7, "Jack Be Nimble, Jack Be Quick," we discuss how rapid your agile analysis can be.

Bernie's Books—An Example in Agile Business Analysis

You are a business analyst and have been hired by Bernie's Books1, shown in Figure 1.2, to improve some of the systems. Bernie's Books is a bricks and mortar bookstore that—against the trend—is thriving while many other bookstores are closing their doors. Bernie's store is located in a well-established shopping street, with restaurants, cafés, and better-quality clothing stores, all of them attracting the type of customer Bernie serves.

Figure 1.2
Bernie's Books is a brick and mortar store selling a better class of books.
(Credit: SpeedKingz/Shutterstock)

Bernie has an idea: he wants to offer customers who buy a hardcopy book the option of a free digital download of the same book.

Bernie also thinks he might start selling his books online. This might be done so that the customer receives an immediate digital download, and the hardcopy book is sent to the customer by mail. This, of course, opens the possibility of digital-only customers.

What Do You Do?

You could rush in and start working on the assumed solution: the automatic digital download. Or you could apply a little analytical thinking to Bernie's problem.

The problem is that we often don't know what the problem is.

FOOTNOTE 1
Readers of Lawrence Block's burglar series may have guessed that our Bernie, the bookstore owner, has been named for Bernie Rhodenbarr, Block's larcenous but lovable bookseller. We thank Mr. Block for his character's name and his entertaining novels.

Bernie has provided a solution: free digital downloads. However, he hasn't said what the problem is. Indeed, we should ask Bernie if he knows what the problem is, and if so, ask if his proposed solutions really solve that problem.

As John Carrol of Penn State College of Information Sciences and Technology puts it, "The worst misstep one can make in design is to solve the wrong problem."

It would not be the first time that someone has put forward a solution without understanding the problem. Also, without knowing more about the context and the environment, we cannot know the extent of the problem.

What's Bernie's Problem?

What is Bernie's need, or what problem does Bernie want to solve? Analytical thinking is partly about finding underlying reasons for things, usually by asking "why?" Suppose we start by asking Bernie this: "Why are you able to run a successful bookshop at a time that other brick and mortar bookshops are closing their doors?" Bernie answers:

"My bookshop is successful because I have enough loyal customers who enjoy reading. They like reading books that appeal to more intelligent people, they are more literary than the general population, and most of all, they want to read the kinds of books that I want to have in my shop. I am not a mass market seller—I don't carry the blockbusters, and my popular fiction is aimed at a better class of reader. I am not exactly upmarket—more "upbrain." I also carry better children's books and a section of cookbooks. This brings the right kind of people into my store. My clientele is pretty loyal.

"Of course, I have to keep attracting a younger audience with the same reading tastes. I cannot keep selling to the same audience forever—they are getting older and at some stage won't be mobile enough to get to my bookstore at all.

"My reason for wanting the addition of the digital download is to appeal to younger people—digital because young people are more likely to read books on their phones or tablets. I also want things to be more convenient for my traditional customers who want the convenience of tablet or phone reading while on a plane or when they don't have room in their handbag or briefcase for the hardcopy.

"I have also seen people come into my bookshop and browse through my books. When they find something they like the look of, they whip out their phone and buy it online from Amazon or one of the other online stores. I call these people "book cover bandits." I lose a sale and see no reason why they shouldn't buy their digital version from me. They're in my bookstore after all."

So now you know a little more and can see that Bernie's assumed solutions are not necessarily going to work.

People Assume They Know the Solution

They don't. They can't know the solution until they know the problem.

You could provide a free e-book version with each paper book bought, but there is nothing yet to say that customers would find any value in having both versions.

You could build a *BernieReader* app, but would anybody use it? And if they did, would it provide value to Bernie?

You could provide free instore Wi-Fi, but there is no evidence that customers would pay attention to it. Alternatively, it might encourage more people to order one of Bernie's books from an online retailer. Additionally, it might give the wrong image if too many people are wandering around using computers or handheld devices in Bernie's store.

Could an online solution harm Bernie's business? It's not unheard of.

Any solution you deliver must solve the right problem; it's the only way you can deliver value. That means that we need to turn our attention to finding the right problem. Once we understand that, we can find the right solution to it. Until then, let's keep an open mind.

Analytical Thinking

To illustrate business analysis agility and analytical thinking, let's take a brief look at the approach we would use for Bernie's Books. We shall go over the activities in detail in subsequent chapters, but for the moment, a quick run-through is sufficient. It is also appropriate; what we are proposing is usually done quite quickly.

Why do we need analytical thinking? Because we need to think rather than be led along by dogma and knee-jerk reactions. For example, one unfortunate piece of dogma is that the team should deliver only the minimum workable product and only the product that has been asked for.

That is a complete abdication of the responsibilities of solution development. It turns the development team into automatons who mechanically transform whatever is asked for (which is often wrong) into software. It is not how good solutions are developed.

Delivering only a minimal working product does not guarantee that anybody will want to use it. You can deliver what they ask for, but the truth is that the customers do not always know what they want, what they need, and what they can get. Nor does the development team.

To deliver *value* by solving the real problem, you need to discover what is needed and what is valuable and not just be a robotic translator.

***The right solution delivers the best value. Nothing less.***

The right solution delivers the best value. Nothing less. To deliver the right solution, the customers and the development team work together, discover the right business problem (it's usually subtler than people think), and then derive the solution that solves this problem in the best possible manner. The best solution has qualities other than minimal. It is convenient, appealing, secure, and robust. It is not—definitely not—"only what is asked for."

Bernie's Business Goals

Why is Bernie making an investment in this project? What are his business goals? Bernie wants to sell books, but more importantly he wants to keep selling books in the future. This means his objectives must include keeping the loyal customers happy, attracting a younger generation of readers, and convincing the book cover bandits to buy from him.

Of course, Bernie can only achieve his goals with the cooperation of his customers. In turn, the customers will only cooperate if there is some benefit to them. This means that we investigate the customers to learn what it is that they value. Once we know that, the problem

shifts to deriving a solution that the customers find valuable and satisfies Bernie's business goals. This is illustrated in Figure 1.3.
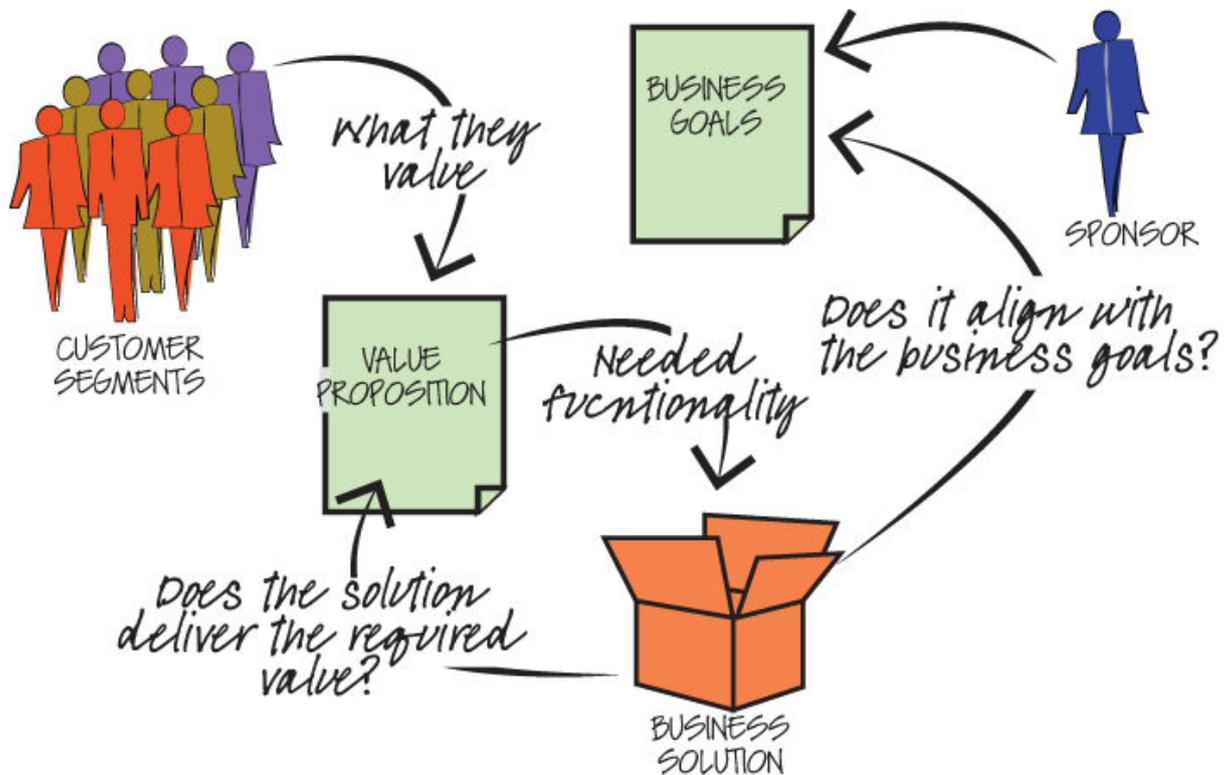
Figure 1.3

Each distinguishable customer segment values something; you can describe it by writing a value proposition. You derive a solution to deliver this value, and this solution must contribute to meeting the business goals.

Customer Segments

In the beginning, pretty much all your client knows is the target audience for the proposed solution. You don't know yet what you will deliver, but you do know who you will deliver it to. You can categorize the target audience using *customer segments*.

A customer segment is a group of people who have a unique profile—a collection of characteristics that separate them from other customer segments in a meaningful way.

For external customers, we suggest you start by segmenting according to the customers' needs. You could also use demographics, attitudes, and tasks that they want to accomplish, and so on.

In the case of in-house consumption, a customer segment is a group of users who share the same profile. The people who inhabit the segment have a set of needs and wants and concerns that set them apart from the other segments. You might prefer to think of this category as *user segments*.

You might also at this stage begin to develop *personas* for your segments. A persona is a virtual representation of the archetypical customer within the segment. The persona is usually a synthesis of typical behaviors, skills, objectives, and so on, and it is usually given enough

human touches—a name, photograph, personal characteristics—to make it real for the team. We have more to say about customer segments and personas in the next chapter.

You will probably find more accurate problems and better solutions if you segment Bernie's customers by their needs and their propensity to buy. This project is about keeping and growing customers, so understanding their buying attitudes seems a reasonable place to start. This thinking gives you three main segments, which Bernie names *loyal customers, twentysomethings*, and *book cover bandits*. There might be some other, smaller segments—people buying a gift, casual browsers, etc.—but for the moment let's focus on the three most valuable segments.