3
ARCHITECTURE
Principles for Designing a Successful Platform

How do we build a platform that invites participation and creates significant value for all its users? How do we provide tools and services that make it easy for producers and consumers to interact in mutually rewarding ways? And how do we design a technological infrastructure that is capable of scaling rapidly, and encouraging positive network effects while minimizing negative ones?

These are daunting challenges. Platforms are complex, multisided systems that must support large networks of users who play different roles and interact in a wide variety of ways. An industry-wide platform—a platform for the health care industry, for example—may need to facilitate interactions among an enormous range of industry participants with motivations that vary widely and change frequently as economic, regulatory, and technological circumstances evolve.

Designers and builders of any complex system often find it difficult to identify a logical starting point. This problem is particularly acute with platform businesses, since they are less familiar and much more complicated than pipeline businesses, which generally feature a straightforward linear design. The natural tendency of those charged with creating a new platform business is to study similar implementations and imitate them. But because no two markets are identical, this strategy often fails. A poorly designed platform produces little or no value for users and generates weak network effects, or none at all.

So where do we start in designing a new platform? The best way is to focus on the fundamentals. What exactly does a platform do, and how does it work?

As we've seen, a platform connects producers with consumers and allows them to exchange value. Some platforms allow direct connection between users, as we see in the case of social networks. These connections then lead to the exchange of value between users. Other platforms do not facilitate direct connections between users but establish other mechanisms for value exchange. For example, on YouTube, videos created by producers are delivered to consumers without a direct connection being made between them.

In this respect, interactions on a platform resemble any economic or social exchange, whether it occurs in the real world or in the virtual world of the Internet. In every such exchange, the producer and the consumer exchange three things: information, goods or services, and some form of currency.

Exchange of information. Whether it's the cattle auctioneer shouting out prices to an assembled crowd of ranchers or an eBay search results page displaying the goods available, every platform interaction starts with the exchange of information. This information enables the parties to decide whether, and how, to engage in any further exchange.

Thus, every platform business must be designed to facilitate the exchange of information. Some platforms have the exchange of information as their sole purpose—for example, a news forum like Reddit or a question-and-answer site like Quora. But even platforms whose primary goal is to enable the exchange of physical goods or services must enable the exchange of information. Uber provides information about driver availability and location in response to passenger requests; Yelp provides information about restaurants to enable users to choose a place to eat; and Upwork allows companies and freelancers to exchange information about themselves to facilitate hiring decisions.

Notice that, in every case, the exchange of information takes place through the platform itself. In fact, this is one of the fundamental characteristics of a platform business.

Exchange of goods or services. As a result of the information exchange, the platform participants may decide to exchange valuable goods or services as well. In some cases, the exchange of goods or services may also occur through the platform. On Facebook, photos, links, and posts with personal or other news are exchanged among users, while on YouTube videos are exchanged. Each item exchanged among platform users can be referred to as a value unit. In some cases, platforms feature sophisticated systems that make it easy and convenient for value units to be exchanged. Upwork, for example, provides clients with built-in tools to manage remote service delivery, so digital goods created by a freelancer, like slide decks and videos, can be exchanged directly through the platform itself.

In other cases, goods or services are exchanged outside of the platform (although information about the delivery may be tracked and exchanged on the platform). Transportation services requested via Uber are delivered on real city streets using actual cars; dinner reservations made via Yelp result in physical meals consumed around real tables in actual restaurants.

Exchange of currency. When goods or services are exchanged between platform participants, they are typically paid for using some form of currency. In many cases, this is traditional currency—money transmitted in one of a variety of ways, including credit card data, a PayPal transaction, a Bitcoin transfer, or (rarely) physical cash.

However, there are other forms of value, and therefore other ways in which consumers "pay" producers in the world of platforms. Video viewers on YouTube or followers on Twitter pay the producer with attention, which adds value to the producer in a variety of ways. (If the producer is a political pundit or business leader, for example, he gains value in the form of growing influence as a thought leader; if she is a singer, actor, or athlete, she gains value in the form of a growing fan base.) Community members on sites like TripAdvisor, Dribbble, and 500px pay by enhancing the reputation of producers whose work they like. Thus, attention, fame, influence, reputation, and other intangible forms of value can play the role of "currency" on a platform.

Sometimes the exchange of currency takes place through the platform itself. This is usually the case when the currency takes the form of attention or reputation. But monetary payment may also take place on the platform, even when the exchange of goods or services occurs elsewhere. Uber and Airbnb, for example, allow service delivery outside the platform, but they close the loop by ensuring that money is exchanged through the platform.

As we'll discuss in more detail in chapter 6, a platform's ability to monetize the value of the exchanges it facilitates is directly related to the types of currency exchange it can capture and internalize. A platform that can internalize the flow of money may be well placed to charge a transaction cut—for example, the fee of 10 percent of the sale price typically charged by eBay after a successful auction. A platform that can capture only attention may monetize its business by collecting payments from a third party that considers the attention valuable—for example, an advertiser willing to pay Facebook for "eyeballs" attracted by posts related to a particular topic.

The platform's goal, then, is to bring together producers and consumers and enable them to engage in these three forms of exchange: of information, of goods or services, and of currency. The platform provides an infrastructure that participants plug in to, which provides tools and rules to make exchanges easy and mutually rewarding.

THE CORE INTERACTION: THE WHY OF PLATFORM DESIGN

Platforms are designed one interaction at a time. Thus, the design of every platform should start with the design of the core interaction that it enables between producers and consumers. The core

interaction is the single most important form of activity that takes place on a platform—the exchange of value that attracts most users to the platform in the first place. The core interaction involves three key components: the participants, the value unit, and the filter. All three must be clearly identified and carefully designed to make the core interaction as easy, attractive, and valuable to users as possible. The fundamental purpose of the platform is to facilitate that core interaction.

This basic rule about the primacy of the core interaction applies despite the fact that many platforms involve a wide range of participants who may interact in a variety of ways. LinkedIn, for example, enables multiple interactions. Professionals exchange ideas about career and business strategy; recruiters exchange information about job listings with potential applicants; human resource managers exchange news about labor market conditions; and thought leaders offer their views about global trends. These various forms of interaction were built into the platform over time, each designed to meet a particular platform goal and to help users create a new form of value. The multisided LinkedIn platform we see today was first designed around a single core interaction: professionals connecting with other professionals.

So let's consider the three key components of the core interaction and how they connect to create value on the platform.

The participants. There are fundamentally two participants in any core interaction: the producer, who creates value, and the consumer, who consumes value. When defining the core interaction, both roles need to be explicitly described and understood.

One nuance of platform design is recognizing that the same user may play a different role in differing interactions. The same person may be both a host and a guest on Airbnb, although he or she will typically perform only one of those roles in a particular interaction. On YouTube, users may upload videos as well as view them. A well-designed platform makes it easy for users to move from role to role.

Conversely, many users, and many types of users, may perform the same role in an interaction. For example, one of the most common interactions on Facebook is a "status update"—a content posting that informs participants in the network about what a particular member is doing or thinking. The producer who drives a change of status on a particular Facebook page may be an individual, a business, a group of friends, or a nonprofit organization, but the fundamental role remains the same. Similarly, videos on YouTube are created by media companies as well as individuals. The incentives that encourage different parties to participate are different, but the roles remain consistent.

The value unit. As we've noted, every interaction starts with an exchange of information that has value to the participants. Thus, in virtually every case, the core interaction starts with the creation of a value unit by the producer.

Here are a few examples. On a marketplace like eBay or Airbnb, the product/service listing information is the value unit that is created by a seller and then served to buyers based on their search query or past interests. On a platform like Kickstarter, the project details constitute the value unit that enables potential backers to make a decision whether to fund it. Videos on YouTube, tweets on Twitter, profiles of professionals on LinkedIn, and listings of available cars on Uber are all value units. In each case, users are provided with a basis for deciding whether or not they want to proceed to some further exchange.

The filter. The value unit is delivered to selected consumers based on filters. A filter is an algorithmic, software-based tool used by the platform to enable the exchange of appropriate value units between users. A well-designed filter ensures that platform users receive only value units that

are relevant and valuable to them; a poorly-designed lter (or no filter at all) means users may be flooded with units they find irrelevant and valueless, which may drive them to abandon the platform.

A search query is an example of a filter. Participants search for information of interest to them by specifying particular search terms: “hotels in and around Hana on the island of Maui” or “single straight males aged 18–25 in Austin TX,” for example. Out of millions of value units previously created by producers (such as hotel proprietors and users in search of a mate), the platform employs the filter to select specific units that match the search terms and delivers them to the consumer.

In one way or another, every platform makes use of filters to manage the exchange of information. Uber’s drivers announce their availability on the platform by sharing various parameters on location, occupancy, and so on—value units that enable them to be matched to the right consumers. When a consumer pulls out her phone and requests a car, she sets up a filter based on her location at the time of the request. Information about the drivers most relevant to the consumer is then exchanged.

Once this exchange of information happens, everything else clicks into action. The car turns up, the traveler is taken to her destination, the appropriate funds are transferred out of the traveler’s account, and the driver is compensated. The core interaction is completed—value has been created and exchanged.

Some platforms have more complicated models, but the basic structure remains the same:

Participants + Value Unit + Filter → Core Interaction

Google’s search engine acts in a fundamentally similar way. Google’s crawlers search the web, creating web page indices (value units). A consumer types in a query. Google combines the query with other specified inputs, such as social signals—the volume of “likes,” retweets, comments, and other responses received by a particular posting on the Internet. This combination of inputs constitutes the filter, which determines which value units are delivered to the consumer.

On Facebook, your entire network creates status updates, pictures, comments, links, and so on—all of them value units that are being added to the platform. Your news feed algorithm, based on signals you’ve given in the past through your interactions with previous content, acts as the filter that determines which units are delivered to you and which are not. When designing a platform, your first and most important job is to decide what your core interaction will be, and then to define the participants, the value units, and the filters to make such core interactions possible.

As we see in cases like LinkedIn and Facebook, platforms often expand over time to embrace many kinds of interactions, each involving different participants, value units, and filters. But successful platforms begin with a single core interaction that consistently generates high value for users. A valuable core interaction that is easy, even enjoyable, to engage in attracts participants and makes the emergence of positive network effects possible.

The crucial role of the value unit. As this description of the core interaction shows, value units play a crucial role in the workings of any platform. Yet, in most cases, platforms don’t create value units; instead, they are created by the producers who participate in the platform. Thus, platforms are “information factories” that have no control over inventory. They create the “factory floor” (that is, they build the platform infrastructure within which value units are produced). They can foster a culture of quality control (by taking steps to encourage producers to create value units that are accurate, useful, relevant, and interesting to consumers). They develop filters that are designed to deliver valuable units while blocking others. But they have no direct control over the production process itself—a striking difference from the traditional pipeline business.1

Fasal is an online system that connects farmers in rural India directly with market agents and other buyers. Via Fasal, farmers can quickly learn the price of goods at a number of nearby markets, choose the sales location most advantageous to them, and use the data to negotiate a better deal, a challenge that exists around the world.2

Sangeet Choudary, one of the authors of this book, led the commercialization and launch of the Fasal initiative. One of the challenges facing Choudary and his team was figuring out what kind of communications infrastructure they could use to enable producers and consumers to share value units. They realized that the big advantage working in their favor was cell phones. More than half of Indian farmers, even the poorest, own and use cell phones. In fact, as in much of the developing world, cell phone use in rural India has spread rapidly. Cellular telephony, with its instant communications capability, became the conduit for the market data the small farmers so desperately needed.

But creating the crucial value units needed to make exchanges possible between the farmers and the mandis (the local market-makers) would prove to be an even more significant challenge. “We needed information of various kinds,” Choudary explains:

Of course, we needed price data from the mandis—current market rates for various grades of commodities ranging from carrots and cauliflower to beans and tomatoes. This turned out to be fairly easy to gather. Some of the agents themselves would provide the information to us. And we supplemented this source by hiring local people to visit each of the mandis to gather price quotes firsthand and report them to us.

The other side of the equation was more difficult. In order to create an electronic information resource that would be truly useful to the farmers, we needed data about the farmers themselves—the crops they were planting, the expected harvest cycle, the locations of the farms, their access to various mandis, and so on. All of these factors would influence the best deal they could make in the marketplace.

But gathering this information from a widely scattered collection of farmers—most of them illiterate—was very tricky. We conducted a series of experiments. We tried relying on word of mouth to spread news about the service we were creating and to collect information for us. We tried using the local “head men”—the unofficial mayors of the villages—as conduits for the information. We tried arranging deals with local sellers of seeds, fertilizer, and SIM cards for cell phones, all of whom were frequently in touch with individual farmers. But none of these methods worked well—the people we tried to work with weren’t interested, and the incentives simply weren’t strong enough to produce a powerful flow of data.

In the end, we had to build our own network of data gatherers—what Indians call a “feet on street” (FOS) sales force. The FOS team went door to door in each village, meeting with farmers and recording key information about their crops and their marketing plans on paper forms. Then they brought the data back to our offices, where we entered it into our spreadsheets. Little by little, we built up the database we needed to begin making sense of the local markets.

As you can see, a focus on the value unit is extremely important if you’re running a platform. Deciding who can create value units, how they are created and integrated into the platform, and what differentiates a high-quality unit from a low-quality one are all critical issues, and we’ll explore these throughout this book.

PULL, FACILITATE, MATCH:THE HOW OF PLATFORM DESIGN

The core interaction is the why of platform design. The whole purpose of a platform is to make core interactions possible—indeed, to the extent possible, to make them inevitable by making them highly valuable to all participants. But how do you achieve this? What can the platform designer do to ensure that valuable core interactions begin to occur in significant numbers, thereby attracting more and more participants to the platform?

In the next few pages, we'll examine the how of platform design. Platforms must perform three key functions in order to encourage a high volume of valuable core interactions, which we summarize as pull, facilitate, and match. The platform must pull the producers and consumers to the platform, which enables interactions among them. It must facilitate their interactions by providing them with tools and rules that make it easy for them to connect and that encourage valuable exchanges (while discouraging others). And it must match producers and consumers effectively by using information about each to connect them in ways they will find mutually rewarding.

All three functions must be performed well if the platform is to succeed. A platform that fails to pull participants will be unable to create the network effects that make the platform valuable. A platform that fails to facilitate interactions—one with clunky technology or overly restrictive policies that make usage difficult—will eventually discourage and alienate participants. And a platform that fails to match participants accurately will waste their time and energy, soon driving participants away.

Let's examine each of these three crucial functions in a little more detail. Effective platform design is all about creating systems that perform these functions as powerfully as possible.

Pull. Attracting consumers to platforms presents challenges that pipeline companies don't face. Consequently, the approach to marketing such platforms is likely to seem counterintuitive, especially to business leaders who grew up in the old pipeline-dominated universe.

To begin with, platforms need to solve a chicken-or-egg problem that pipeline businesses don't suffer from: users won't come to a platform unless it has value, and a platform won't have value unless users use it. Most platforms fail simply because they never overcome this problem. It's such an important challenge that we'll devote all of chapter 5 to analyzing it and helping you solve it.

A second pull challenge revolves around keeping the interest of users who visit or sign up for the platform. The large social networks of our day have all faced this problem at some point. Facebook, for example, discovered that users found the platform valuable only after they had connected to a minimum number of other users. Until then, they were likely to stop using the network entirely. In response, Facebook shifted its marketing efforts away from recruiting new members to helping them form connections.

One powerful tool that encourages users to keep returning to the platform is the feedback loop. A feedback loop in a platform may take various forms, all of which serve to create a constant stream of self-reinforcing activity. In the typical feedback loop, a flow of value units is that generates a response from the user. If the units are relevant and interesting, the user will be drawn to the platform repeatedly, generating a further flow of value units and facilitating more interactions. Effective feedback loops help to swell the network, increase value creation, and enhance network effects.

One kind of feedback loop is the single-user feedback loop. This involves an algorithm built into the platform infrastructure that analyzes user activity, draws conclusions about the user's interests, preferences, and needs, and recommends new value units and connections that the user is likely to find valuable. When it is adroitly designed and programmed, the single-user feedback loop can be a powerful tool for increasing activity, since the more the participant uses the platform, the more the platform "learns" about him and the more accurate its recommendations become.

In a multi-user feedback loop, activity from a producer is delivered to relevant consumers, whose activity in turn is fed back to the producer. When effective, this creates a virtuous cycle, encouraging activity on both sides and ultimately strengthening network effects. Facebook's news feed is a classic multiuser feedback loop. Status updates from producers are served to consumers, whose likes and comments serve as feedback to the producers. The constant flow of value units stimulates still more activity, making the platform increasingly valuable to all participants.

Other factors strengthen or weaken a platform's ability to pull users. One is the value of the currency available for exchange on the platform. As we've discussed, some platform exchanges are paid for in intangible forms of currency: attention, popularity, influence, and so on. Thus, one form of network effect is the increased attractiveness of the currency available on a platform that is growing in size. Because Twitter has achieved such an enormous user base, a successful tweet is likely to attract far more currency in the form of attention than the same message disseminated on some other platform. So Twitter's huge size enlarges its pull, encouraging still more participant activity and making a competitive challenge to the platform increasingly unlikely.

Pull can also be increased by leveraging the outside networks of participants. Instagram and WhatsApp pulled in tens of millions of participants in a few years mainly by piggybacking on their users' Facebook networks. We'll examine these and other techniques for turbocharging pull in more detail in chapter 5, which focuses on the launch process.

Facilitate. Unlike traditional pipeline businesses, platforms don't control value creation. Instead, they create an infrastructure in which value can be created and exchanged, and lay out principles that govern these interactions. That's what the process of facilitating is all about.

One aspect of facilitating interactions is making it as easy as possible for producers to create and exchange valuable goods and services via the platform. This may involve providing creative tools for collaboration and sharing, as the Canadian photography platform 500px does with its infrastructure, which allows photographers to host their entire portfolios on the platform, or as invention platform Quirky does with its tools for letting users work together on creative ideas for innovative products and services.

Facilitating interactions may also involve reducing barriers to usage. Not long ago, a Facebook user who wanted to share photos with friends had to use a camera, transfer the images to a computer, use Photoshop or another software package to edit them, and finally upload them to Facebook. Instagram enabled users to snap, modify, and share pictures in three clicks on a single device. Lowering barriers to usage in this way encourages interactions and helps expand participation on the platform.

In some cases, increasing barriers has a positive effect on usage. Sittercity is platform that helps parents find babysitters. To inspire trust among its users (the parents), Sittercity has imposed a stringent set of rules that restrict those who can sign up as producers (the babysitters). In other cases, platforms must develop intrusive rules for curating value units and other producer-created content in order to encourage desirable interactions and discourage undesirable ones. Though relatively rare, misdeeds like the racist and sexist abuse spewed by trolls on Reddit, the murders of people found through Craigslist, and the trashing of apartments booked through Airbnb illustrate how undesirable interactions damage network effects. Match. A successful platform creates efficiencies by matching the right users with one another and ensuring that the most relevant goods and services are exchanged. It accomplishes this by using data about producers, consumers, the value units created, and the goods and services to be exchanged. The more data the platform has to work with—and the better designed the algorithms used to collect, organize, sort, parse, and interpret the data—the more accurate the

filters, the more relevant and useful the information exchanged, and the more rewarding the ultimate match between producer and consumer.

The data required for optimal matching may be extremely diverse. They range from relatively static information such as identity, gender, and nationality to dynamic information such as location, relationship status, age, and point-in-time interest (as reflected in a search query). Sophisticated data models like the Facebook news feed may build a filter that considers all these factors as well as all of the participant's previous activities on the platform.

As part of the design process, platform companies need to develop an explicit data acquisition strategy. Users vary greatly in their willingness to share data and their readiness to respond to data-driven activity recommendations. Some platforms use incentives to encourage participants to provide data about themselves; others leverage game elements to gather data from users. LinkedIn famously used a progress bar to encourage users to progressively submit more information about themselves, thereby completing their personal data profiles. Data may also be acquired from third-party providers. Some mobile apps, such as the music streaming app Spotify, ask users to sign in using their Facebook identities, which helps the app pull in initial data to use in facilitating accurate matches. However, resistance from some users has led many app makers, including Spotify, to provide alternative ways to sign in that don't require a Facebook link.

Successful platforms create mutually rewarding matches on a consistent basis. As such, continual improvement of data acquisition and analysis methods is an important challenge for any organization seeking to build and maintain a platform.

Balancing the three functions. All three key functions—pull, facilitate, and match—are essential to a successful platform. But not all platforms are equally good at all three. It's possible for a platform to survive, at least for a time, thanks mainly to its strength at a particular function.

As of mid-2015, Craigslist continues to rule the classifieds space despite a poor interface, an utter lack of governance, and an unsophisticated data system. Craigslist's massive network keeps pulling users back. Thus, this platform's enormous advantage in pull has compensated for its weaknesses in facilitate and match—at least, so far.

Vimeo and YouTube coexist in the video sharing arena by focusing on different functions. YouTube employs a strong pull and deep understanding of the use of data in matching, while Vimeo differentiates itself through better hosting, bandwidth, and other tools for facilitating production and consumption.

BEYOND THE CORE INTERACTION

As we've seen, platform design begins with the core interaction. But over time, successful platforms tend to scale by layering new interactions on top of the core interaction.

In some cases, the gradual addition of new interactions is part of the long-term business plan that platform founders had in mind from the beginning. In early 2015, both Uber and Lyft began experimenting with a new ride-sharing service that complements their familiar call-a-taxi business model. The new services, known as UberPool and Lyft Line, allow two or more passengers traveling in the same direction to find one another and share a ride, thereby reducing their cost while increasing the revenues enjoyed by the driver. Lyft cofounder Logan Green says that ride-sharing was always part of the Lyft idea. The initial version of Lyft, he explains, was designed to attract an initial customer base "in every market." Having achieved that, he continues, "Now we get to play that next card and start matching up people to take rides."3

Uber isn't taking the competition lightly. To try to ensure that its ride-sharing service out-competes Lyft's, Uber has joined the bidding for Here, a digital mapping service owned by Nokia that is the chief alternative to Google Maps. Uber hopes to buy Here and use its mapping power to produce swift and accurate ride-sharing matches more effectively than any other service.4

In other cases, ideas for new interactions emerge from experience, observation, and necessity. In its search for new drivers, Uber discovered that many of its best prospects were recent immigrants to the U.S. who were eager to supplement their incomes by driving for Uber but who lacked the credit histories and financial qualifications needed to finance car purchases. Andrew Chapin of Uber's driver operations group came up with the idea of having Uber act as a middleman to guarantee car loans for its drivers, deducting repayments from driver revenue and sending them directly to the lenders. Finance companies like the program because loans backed by Uber's massive corporate cash flow are almost risk-free, and local auto dealers are happy with the additional inventory turnover.5

Another example: LinkedIn started by enabling professionals to network with one another. During its initial days, it focused exclusively on enabling its core interaction. Over time, the team at LinkedIn realized that the platform hadn't created the same high level of daily engagement that Facebook and a handful of other platforms had achieved. To address this issue, LinkedIn layered an additional interaction on top of its core interaction: It began allowing users to organize themselves into groups and start discussions.

This second form of interaction didn't achieve the popularity LinkedIn had hoped for either. Given the self-promotional behavior that a professional network encourages, the loudest users in the groups were often also the most obnoxious. So LinkedIn went on to add a further interaction, partly driven by the quest to monetize the platform: it allowed recruiters to use the site to target candidates, and advertisers to target ads to relevant professionals. Later still, LinkedIn created another interaction when it allowed thought leaders, and subsequently all users, to publish posts on LinkedIn for others to read, effectively turning the site into a publishing platform. This combination of many forms of interaction gives users more reasons to visit LinkedIn.

The evolution of Uber, Lyft, and LinkedIn illustrates several of the ways that new interactions may be layered on top of the core interaction in a given platform:

- By changing the value unit exchanged between existing users (as when LinkedIn shifted the basis of information exchange from user profiles to discussion posts)

- By introducing a new category of users as either producers or consumers (as when LinkedIn invited recruiters and advertisers to join the platform as producers)

- By allowing users to exchange new kinds of value units (as when Uber and Lyft made it possible for riders to share rides as well as arranging solo pickups)

- By curating members of an existing user group to create a new category of users (as when LinkedIn designated certain participants as "thought leaders" and invited them to become producers of informational posts)

Of course, not every new interaction is successful. Jake McKeon founded the social network Moodswing as a place where people could share their emotional states, from elation to gloom. Over time, he found that some users were turning to Moodswing in times of severe depression, and a few even used the site to threaten suicide. Distressed, McKeon decided to try to provide these users with the emotional support they needed. He concocted a plan to recruit psychology students who would volunteer to offer counseling and advice via chat lines to depressed Moodswing members. The

volunteers would be tested and vetted in an effort to curate their quality. This “amateur therapy” offering would represent a new form of value exchange facilitated by Moodswing.

It’s an intriguing concept, but one that raises some obvious questions—in particular the potential danger in having untrained and unlicensed counselors offering psychological guidance to people whose lives are at risk. As of mid-2014, McKeon was in the process of seeking crowdfunding support for the project. It remains to be seen whether Moodswing’s new interaction will be launched successfully and produce the user benefits McKeon hopes for.

APPLYING THE END-TO-END PRINCIPLETO PLATFORM DESIGN

As we’ve seen, adding new features and interactions to a platform can be a powerful way to increase its usefulness and attract more users. But innovation can easily lead to excessive complexity, which makes the platform more difficult for users to navigate. Needless complexity can also create enormous technical problems for the programmers, content developers, and managers who are charged with updating and maintaining the platform. The derisive term bloatware has been coined to describe software systems that have become complicated, slow, and inefficient through thoughtless accretion of features.

However, avoiding innovation altogether is no solution. A platform that fails to evolve by adding desirable new features is likely to be abandoned by users who discover a competing platform with more to offer. Instead, a way must be found to strike a balance, changing the core platform only slowly while allowing positive adaptations at the periphery.

This concept is the equivalent, for a platform business, of a long-established computer networking idea known as the end-to-end principle. Originally formulated in 1981 by J. H. Saltzer, D. P. Reed, and D. D. Clark, the end-to-end principle states that, in a general-purpose network, application-specific functions ought to reside in the end hosts of a network rather than in intermediary nodes.6 In other words, activities that are not central to the workings of the network but valuable only to particular users should be located at the edges of the network rather than at its heart. In this way, secondary functions don’t interfere with or draw resources away from the core activities of the network, nor do they complicate the task of maintaining or updating the network as a whole. Over time, the end-to-end principle has been expanded from network design to the design of many other complex computing environments.

One of the most storied examples of failure to heed the end-to-end principle concerns Microsoft’s 2007 introduction of Vista, the latest version of its Windows operating system. CEO Steven Ballmer trumpeted Vista as “the biggest product launch in Microsoft’s history” and backed the launch with a marketing budget of hundreds of millions of dollars.7

Yet Vista failed badly. The problem was that Microsoft’s design team had sought to retain the software components needed to maintain backwards compatibility with older computer systems while adding features needed by next-generation systems—all within the core platform. As a result, Vista was less stable and more complex than its predecessor, Windows XP, and outside app developers had difficulty writing code for it.8

Critics described Vista as worse than bloatware—in fact, they dubbed it goatware because it ate all a system’s resources.9 To this day, millions of Windows users have refused to adopt Vista, clinging to Windows XP despite repeated efforts by Microsoft to retire it. Ironically, while Microsoft stopped retail sales of XP in 2008 and of Vista in 2010, XP’s market share in 2015 was above 12 percent, while that of Vista was below 2 percent.10

By contrast, when Steve Jobs returned to the leadership of Apple in 1997 after his years developing the ambitious but unsuccessful NeXT computer, he made a crucial decision that honored the end-to-end principle and helped lead to Apple's subsequent success. At NeXT, Jobs and his team had developed an elegant new operating system with a clean, layered architecture and a beautiful graphical interface. Now, planning a successor to Apple's Mac OS 9 operating system, Jobs faced a hard choice: he could merge the NeXT and Mac OS 9 software code, thereby producing an operating system that would be compatible with both systems, or he could jettison Mac OS 9 in favor of NeXT's clean architecture.

Jobs placed a dangerous bet on dumping the old code from OS 9. However, he made one concession: the design team developed a separate "Classic Environment" that would allow consumers to run their old OS 9 applications. This compartmentalized approach satisfied the end-to-end principle. The old code did not slow down or add complexity to the new applications, and new Mac buyers were unburdened by software written to accommodate apps they didn't own. Jobs's choice made innovation on the new Mac OS X easier and more efficient, which enabled Apple to develop new features that made Microsoft's operating systems look dated by comparison.11

The end-to-end concept can also be applied to the design of a platform. In this case, the principle states that application-specific features should reside in the layer of process at the edge or on top of the platform, rather than at the roots deep within the platform. Only the highest-volume, highest-value features that cut across apps should become part of the core platform.

There are two reasons for this rule. First, when specific new features are incorporated into the core platform rather than attached to the periphery, applications that do not use those features will appear slow and inefficient. By contrast, when application-specific features are run by the app itself rather than by the core platform, the user experience will be much cleaner.

Second, a platform ecosystem can evolve faster when the core platform is a clean, simple system rather than a tangle of numerous features. For this reason, C. Y. Baldwin and K. B. Clark of Harvard Business School describe a well-designed platform as consisting of a stable core layer that restricts variety, sitting underneath an evolving layer that enables variety.12

Today's best-designed platforms incorporate this structural principle. For example, Amazon Web Services (AWS), the most successful platform for providing cloud-based information storage and management, focuses on optimizing a handful of basic operations, including data storage, computation, and messaging.13 Other services, which are used by just a fraction of AWS customers, are restricted to the periphery of the platform and provided through purpose-built apps.

THE POWER OF MODULARITY

There are advantages to an integral approach where the system is developed as quickly as possible to serve a single purpose, especially in the early days of a platform. However, in the long run, a successful platform must have a more modular approach. A full discussion of this trade-off is well beyond the scope of this chapter, but we will cover some of the important ideas. We begin with a definition provided by Baldwin and Clark (1996):

Modularity is a strategy for organizing complex products and processes efficiently. A modular system is composed of units (or modules) that are designed independently but still function as an integrated whole. Designers achieve modularity by partitioning information into visible design rules and hidden design parameters. Modularity is beneficial only if the partition is precise, unambiguous, and complete. The visible design rules (also called visible information) are decisions that affect

subsequent design decisions. Ideally, the visible design rules are established early in a design process and communicated broadly to those involved.14

In a 2008 paper, Carliss Young Baldwin and C. Jason Woodard provided a useful and succinct definition of a stable system core:

We argue that the fundamental architecture behind all platforms is essentially the same: namely, the system is partitioned into a set of "core" components with low variety and a complementary set of "peripheral" components with high variety. The low-variety components constitute the platform. They are the long-lived elements of the system and thus implicitly or explicitly establish the system's interfaces, [and] the rules governing interactions among the different parts.15

A critical factor that makes modularity so effective is that when systems are cleanly partitioned into subsystems, they can work as a whole by connecting and communicating through well-defined interfaces. The implication is that subsystems can be designed independently so long as they adhere to overall design rules and connect to the rest of the system only through standard interfaces. Readers will likely have heard the term application programming interfaces, or APIs. These are the standard interfaces that systems such as Google Maps, the New York Stock Exchange, Salesforce, Thomson Reuters Eikon, Twitter, and many more use to facilitate access by external entities to core resources.16

Amazon has been especially effective at opening APIs to its modular services. Figure 3.1 compares the range of APIs made available by Amazon and by the leading traditional retailer, Walmart, which is making a strong effort to become a significant platform competitor. As you can see, Amazon has by far outstripped Walmart in the number and variety of APIs provided. The power of modularity is one of the reasons that the personal computer industry grew so quickly in the 1990s. The key components of PC systems were central processing units (CPUs) that provided the computation, graphical processing units (GPUs) that created rich images on the screen, random access memory (RAM) that provided working storage, and a spinning hard drive (HD) that provided large amounts of long-term storage. Each of these subsystems communicated with the others using well-defined interfaces that allowed for tremendous innovation, as firms such as Intel (CPUs), ATI and Nvidia (GPUs), Kingston (RAM), and Seagate (HD) all worked independently to improve the performance of their products The reason that most platforms launch with a tightly integrated architectural design is that there is significant work involved in carefully specifying subsystem interfaces—and even in simply documenting them. When firms are pursuing narrow market windows with limited engineering resources, they can easily be tempted to skip the hard work of decomposing systems into clean modules and instead proceed as quickly as possible to a viable solution. Over time, however, this approach makes it much more difficult to mobilize an external ecosystem of developers who can build on top of the core platform and extend its offerings into new markets.18 Thus, a firm that has an integral architecture will likely have to invest in remaking its core technology.19

RE-ARCHITECTING THE PLATFORM

It is possible to pull off the trick of re-architecting a system toward a modular design. The first step is to analyze the degree of modularity the system has already achieved. Fortunately, a number of tools have been developed to accomplish this goal. Key among these are "design structure matrices" that allow a visual examination of the dependencies in complex systems.20

In a 2006 article in Management Science, Alan MacCormack and Carliss Baldwin document an example of a product that successfully evolved from an integral to a modular architecture.21 When the software was put into the public domain as open source, the commercial firm that owned the copyright invested significant resources to make the transition. This was critical because the software

could not have been maintained by distributed teams of volunteer developers if it had not been broken into smaller subsystems.

The need to re-architect a complex system is not unique to software. In the early 1990s, Intel faced a major challenge in growing its market. The performance of Intel's CPU chips was doubling every eighteen to twenty-four months.22 Similar performance improvements were occurring in the other key PC subsystems: GPU, RAM, and hard drives. However, the information connections between the subsystems were still defined by an old standard called the Industry Standard Architectture (ISA). As a result, consumers saw little improvement in PC performance and thus had little reason to buy new machines. In a 2002 paper, Michael A. Cusumano and Annabelle Gawer document how Intel took the lead by investing in a new Peripheral Component Interconnect (PCI) to better connect the main subsystems, and the universal serial bus (USB) standard which fostered tremendous amounts of innovation in connected devices such as computer mice, cameras, microphones, keyboards, printers, scanners, external hard drives, and much more.23

ITERATIVE IMPROVEMENT: THE ANTI-DESIGN PRINCIPLE

When you're launching a new platform—or seeking to enhance and grow an existing platform—thoughtful attention to the principles of platform design will maximize your chances of value creation.24 But as we've seen, platforms cannot be entirely planned; they also emerge. Remember that one of the key characteristics that distinguishes a platform from a traditional business is that most of the activity is controlled by users, not by the owners or managers of the platform. It's inevitable that participants will use the platform in ways you never anticipated or planned.

Twitter was never meant to have a discovery mechanism. It originated as simply a reverse-chronological stream of feeds. There was no way to seek out tweets on particular topics other than by scrolling through pages of unrelated and irrelevant content. Chris Messina, an engineer at Google, originally suggested the use of hashtags to annotate and discover similar tweets. Today, the hashtag has become a mainstay of Twitter.

Platform designers should always leave room for serendipitous discoveries, as users often lead the way to where the design should evolve. Close monitoring of user behavior on the platform is almost certain to reveal unexpected patterns—some of which may suggest fruitful new areas for value creation. The best platforms allow room for user quirks, and they are open enough to gradually incorporate such quirks into the design of the platform.

Smart design is essential to building and maintaining a successful platform. But sometimes the best design is anti-design, which makes space for the accidental, the spontaneous, and even the bizarre.25

TAKEAWAYS FROM CHAPTER THREE

- The design of a platform should begin with its core interaction—one kind of interaction that is at the heart of the platform's value-creation mission.
- Three key elements define the core interaction: the participants, the value unit, and the filter. Of these, the value unit is the most crucial, and often the most difficult to control.
- In order to make the core interaction easy and even inevitable, a platform must perform three crucial functions: pull, facilitate, and match. All three are essential, and each has its special challenges.
- As a platform grows, it often finds ways to expand beyond the core interaction. New kinds of interactions may be layered on top of the core interaction, often attracting new participants in the process.

- It’s important to design a platform thoughtfully to make mutually satisfying interactions easy for large numbers of users. But it’s also important to leave room for serendipity and the unexpected, since users themselves will find new ways to create value on the platform.