INTRODUCTION

Agile is often described as iteratively building software in increments. This is a focus on the team and the mechanics of how they work. It is more effective to focus on the reason for being Agile – achieving business agility. Business agility is the ability to deliver highest business value quickly, predictably, sustainably and with high quality. Building software is not our goal, realizing business value is. Software is often a component of this, of course.

Faster “time-to-market” is a well-known mantra. The common trap is to try to achieve time-to-market simply by trying to go faster. The better approach is to focus on what will truly provide the greatest value. Instead of trying to go faster, build those chunks of business value which are most important. Go smaller and focus on those value adds that contain the most value for their efforts.

We call this the “Minimum Business Increment” (MBI). The MBI is a key to delivering value quickly by focusing on the most important business value to realize. It becomes a cornerstone concept in planning and alignment.

FREQUENCY OF DELIVERY VS. TIME TO DELIVER

All too often, organizations cannot deliver value quickly. While deliveries may take place on a quarterly basis, the work being delivered has often been under development for a year. The issue is not how frequently you deliver but rather what is the time frame from start to finish.

DELIVERING IN STAGES

To set the stage for incremental delivery of business value, it is important to understand why this is so beneficial. Mark Denne and Cleland-Huang put forth some compelling arguments for quick, frequent delivery in their book, Software by Numbers: Low-Risk, High-Return Development.

Figure 1 illustrates the economics of delivery they propose. This diagram illustrates how, at the beginning of a software project, the organization loses money during the investment period. After the product is released, there is a payback period and that leads to making enough money to incur profit.

Consider an example. Suppose you are the CEO of a company that does product development projects for other companies. You have received a Request for Proposal (RFP) that requires 100 features to be done in 10 months with a certain amount of quality and that the price is fixed. Here is what the RFP stipulates.

- They must be of the quality specified, not better or worse.

How could you win such a contract? Perhaps if your company had a better track record than the competition, you could win on the basis of credibility; however, let’s suppose that is not the case. How else could you compete?

Consider the rate of delivery.
For example, suppose you commit to delivering 50 features after five months and the other 50 features at the 10 months. You are complying with the requirement to deliver 100 features after 10 months; you are just delivering some earlier. What would this look like economically? Assume delivering the first half of the features cost you half the work and delivered half the value, the investment and return graphs would look something like that shown in Figure 2. Even if these assumptions are not completely correct, consider this simple model and the issues that must be dealt with to make it more realistic.

While you cannot always do this staged release scheme, most software can be delivered in increments. For example, ignore the factors of extra development time for a staged release (which would make the cost curves deeper) as well as extra delivery and consumption costs for a staged release (also making the cost curves deeper). Figure 3 shows the combined cost curves in Figure 2 compared with the cost curve of the single release, the net cost of the stage release is less because the return from the first, partial release, offsets the investment required by the second partial release.

Figure 4 shows the total return that is achieved. Figure 5 compares this phased return with the return from a single release. The top line is the return from the staged releases while the line underneath it is the return from the single release. The reason the investment for the staged release is less is that while the second release is being built the first release is giving us a return.

While this may have been a contrived example, it illustrates the value of delivering in stages: getting value sooner. In the real world, after the first release we could take advantage of what we learned and pivot if needed. The key is to recognize that the three dimensions often quoted (scope, cost and time) leave out the fourth dimension of iterative delivery.

TARGETED MARKETS AND INCREASED ALIGNMENT

When considering the ‘Minimum Business Increment’ to select, it is useful to look at the different scenarios in which the new capability can be used. Very often each of these scenarios will have different market segments using them. MBIs can be selected on the basis of which scenarios and which markets should be targeted for maximum value delivery.

There is a powerful side effect to doing this. By creating MBIs that represent focused business value, they can be sequenced in order of importance to the organization. That is, those MBIs that deliver the greatest value soonest can be sequenced as a higher priority than those that don’t. This alignment of what is of greatest business value can also be
used to align disparate teams in building things in this same order, thereby working together more effectively.

Note that MBIs are fundamentally different from epics. First, the Business typically does not know or care what an epic is. This is with good reason; for example, you can have strong feelings about a car but never care about how fuel injection works. An epic is simply an Agile construct that represents a “big story” without connection to value. MBIs are oriented toward business stakeholders and are tied to business value. An MBI is an atomic unit of value and this enables it to provide deeper agreement on if and when it should be built.

THE ISSUES OF STAGED DELIVERY

Of course, it is not as simple as this. Building software in stages may take more effort than building it in one pass. Two deliveries may cost more and may be more disruptive for customers. These factors would all take the return curves down by adding more cost. Let’s look at the concerns about extra costs and then consider the advantages of phased deliveries.

EXTRA COST OF BUILDING PHASED DELIVERIES

While this is of concern to many people, it does not have to cost more to building stages. If you are extending a legacy system that has large amounts of technical debt and is difficult to test, then multiple deliveries may, in fact, cost more. But if your code quality is good and you have automated tests, it often costs less to build and deliver in stages. There are several reasons that contribute to this. One is phased deliveries can result in not building a certain function that at first appears necessary, but with the feedback from the first phase, you learn that it is not as desirable as first thought. Also, the use of emergent design can actually speed up phased delivery. Emergent design is beyond the focus of this book. For more information, see Scott Bain’s excellent book, *Emergent Design: The evolutionary nature of professional software development.*

EXTRA COST OF DEPLOYMENT

This can result in extra cost in deployment; however, it does not necessarily have to be so. Making an investment in being able to release and deploy quickly is important so as to take advantage of the extra return it affords.

EXTRA COST OF CONSUMPTION

While there are some applications in which users do not want updated on a regular basis, studies have shown that most users prefer small, frequent upgrades to large, infrequent ones. The salient characteristic, of course, is the level of disruption to the user. If the consumption by the user is small enough, they won’t mind it. Most objections to upgrades are due to the effort required by the user to do the upgrade.

THE POTENTIAL RETURN OF STAGED DELIVERY

Of course, in the real world, when you can make staged deliveries, you will not arbitrarily pick half the features and release them. Instead, you would pick those features that made the most sense to deliver. This would include looking at the factors such as where is the greatest value and what will it cost to build, deploy, and consume the release. Clearly, the smallest release is useful to the customer and worth the incremental cost of building and deploying is what you should do. We prefer to call this the “Minimum Business Increment” or MBI. Others have used the terms Minimal Marketable Feature (MMF), Minimal Viable Feature (MVF), and Minimal Viable Product (MVP).

These are not just name changes. They also indicate intent of the phase. Eric Ries talks about MVPs as a way to test the market—both to see if your product is viable and to gain quick entry. While we talked about how the return curves for a phased release may not be as good as illustrated, there are some other considerations that may make them considerably better. In a highly competitive situation, for example, whoever gets into the marketplace first may capture it.

In this extreme case, gaining entry into the market three months before our competition may enable our client to own it. Coming in two months after their competition may be an effective barrier of entry. Another advantage is that customers will be able to provide greater feedback to improve the effectiveness of the subsequent phases of delivery.

LOWERING RISKS IN DEVELOPMENT

The biggest risk in software development has long been building the wrong thing. Or overbuilding the right thing (which basically means some of what was built was right

**MMF VS. MVP VS. MBI**

The concept of MMF was developed Denne and Cleland-Huang. We renamed MMF to MBI to emphasize that it was defined from how the business was driving for value in meeting the needs of its customers. In any event, MMF has since been redefined by SAFe®. The original definition of MVP prior to the Lean-Startup is also consistent with MBI. However, as redefined by Eric Ries, MVPs are about discovering what is of value to clients that are early adopters. While they can be of use to large organizations, they are intended when new products are being created, not when existing products are enhanced.
and some of it was not as useful. MBIs enable a new type of evolution of products which can dramatically lower the risk of building features of little value.

**SYSTEM EVOLUTION**

Many applications are built by layer. First a framework is built and then different layers are added until it is complete. We call this “system evolution.” This is shown in Figure 6.

In system evolution, the system is built in phases, one on top of another. These can be done in parallel as well but until all phases are done there is no value.

**BUSINESS EVOLUTION**

In “business evolution,” you deliver increments of value as soon as possible. This is accomplished by using MBIs. The purpose of staged deliveries is to deliver real value sooner and to stop delivering when enough value has been delivered. Business evolution delivers a series of business increments as shown in Figure 7.

Here are some of the advantages business evolution offers over system evolution.

- Value is realized sooner.
- Customers get to provide feedback quickly on what they want.
- The development team can take advantage of what they learned from early releases.
- Releases of a new product can be delayed when enough value is achieved.
- The risk of spending significant effort on work that does not produce value is lower.

The second bullet point is important. It is a truism that customers often do not know what they want until they see something... or until after they see what it is they don’t want!

**LOWERING RISK**

In both system evolution and business evolution, it is possible to build the wrong thing but the impact is much different. Figure 8 shows what happens in system evolution if the wrong thing is built at the end. Business evolution does not guarantee that wrong things won’t be built, but problems will be discovered much earlier (see Figure 9).
Comparing the Risks of the Two Approaches

Consider the risks involved in system evolution and business evolution. Figure 10 illustrates the potential risk curves of each approach. Both start with the same risk and both end with no risk (that is, when it is finished, everything is finished and there is no more risk involved). Compare the risk curves in each approach. What are the implications if risk and uncertainty start out high and stay high until the end or go down at a steady pace over time or go down quickly?

In system evolution, the risk comes from not getting good feedback from the customer until the very end. You run the risk of building the wrong thing, of over-building the system, or even of building the architecture for the system incorrectly. By contrast, in business evolution, you find out early if something is going wrong and you can make adjustments. As Figure 11 shows, in business evolution the risk goes down significantly with the first release.

![Figure 10. The possible risk curves for system and business evolution](image)

![Figure 11. System evolution and business evolution](image)

Opportunities and the Need to Look Across Products

Both XP and Scrum bring a focus on the interaction between the team and the customer. This is appropriate when teams are independent and work in only one area. In larger organizations, a broader view is needed. Within that broader view, the focus of the team to the customer is good so long as the context is not ignored.

To understand this, consider the different possible rate of returns you can have on projects. The ideal case is what is called the “Pareto Curve.” This occurs when 80% of the value is achieved by doing 20% of the work. The challenge is that Product Owners are tasked with continuously building the product they are responsible for. With system evolution there is a tendency to build all that has been asked for. But with business evolution the Product Owner can see if most of the gains have been achieved and that a new project should be started. For example, in Figure 6 it is possible that another project could be worked on after the fourth release.

Although customers who are using this product may want you to continue working on this, you run the risk of missing opportunities for other products that can give a big return on its first release. In other words, instead of getting the tail end of the Pareto Curve, you may want to look for other products that will give a big return for relatively little effort.

We often see budgeting errors in this situation. For example, companies continue to provide funding for established products because they base budgets along the lines the returns are occurring in. The better approach is to have budgets made along the lines of what future returns would be available. Of course, it is important to consider what will happen to existing systems if they are not enhanced over time. But even this involves looking at investment on the basis of future returns rather than on past laurels or on the idea that a team belongs to a customer.

Looking forward in this way opens up the opportunity for better product portfolio management. You can pick those features, regardless of which product they are for, that produces the greatest value to the organization.

Finally, it is good to note that not all products give such a Pareto return. Many give fairly linear results with each release providing an amount of value proportional to the effort it took to achieve it. Some product enhancements return value relatively quickly; others take a few months to build. You want to avoid long delays in realizing value simply because you did not consider the potential of incremental business delivery.

Al Shalloway is the founder and CEO of Net Objectives. With over 40 years of experience, Alan is an industry thought leader in Lean, Kanban, product portfolio management, Scrum and agile design. He helps companies transition to Lean and Agile methods enterprise-wide as well teaches courses in these areas. Alan has developed training and coaching methods for Lean-Agile that have helped Net Objectives’ clients achieve long-term, sustainable productivity gains. He is a popular speaker at prestigious conferences worldwide. He is the primary author of . Alan has worked in literally dozens of industries over his career. He is a co-founder and board member for the Lean Software and Systems Consortium. He has a Masters in Computer Science from M.I.T. and a Masters in Mathematics from Emory University.

Follow Al on twitter @ alshalloway
THE LEAN STARTUP MOVEMENT

It is worth taking some lessons from the Lean Startup movement here. I’ve been discussing return of business value of mostly established products. The Lean Startup movement, however, suggests delivering business value incrementally in order to determine which products actually have value. In other words, delivering part of a system, that may not be complete yet but which will provide some value to customers while indicating how much value the product can eventually provide is a useful consideration. The Lean Startup approach using Minimum Viable Products (MVPs) is a different, albeit useful, tact on incremental business value.

Both MBIs and MVPs have the same heritage of Denne and Cleland-Huang’s MMF. Having both in your arsenal of tools can be quite powerful. But it is important to know the difference. MVPs as described in Lean Startup are designed for startups and are focused around the discovery of new product value. MBIs are designed for all companies of any maturity and is focused on increasing value realized by the business. Both allow for pivoting as discovery of the true value of the product is delivered.

MORE RESOURCES

Visit the Net Objectives Portal at https://portal.netobjectives.com/whitepapers to see these related articles:
- Net Objectives approach to SAFe®
- Why tailored Agile transformations are more effective, less expensive and less risky

NET OBJECTIVES

We are committed to delivering the principles, practices, and perspectives that businesses must know in order to maximize their return on their technology solution and software development efforts. We combine our experience and a time proven approach based on lean thinking to continuously extend the capability of what is possible in creating effective technology delivery organizations (IT or product). We provide these learned methods to our clients to assist them in achieving their goals and in assisting them in making their organizations more successful.

A full set of papers and articles may be found at https://portal.netobjectives.com/whitepapers

Figure 12. Comparing system evolution and business evolution
BUSINESS-DRIVEN SOFTWARE DEVELOPMENT

Business-Driven Software Development is Net Objectives’ proprietary integration of Lean-Thinking with Agile methods across the business, management and development teams to maximize the value delivered from a software development organization. This approach has a consistent track record of delivering higher quality products faster and with lower cost than other methods.

Business-Driven Software Development goes beyond the first generation of Agile methods such as Scrum and XP by viewing the entire value stream of development. Lean-Thinking enables product portfolio management, release planning and critical metrics to create a top-down vision while still promoting a bottom-up implementation.

Our approach integrates business, management and teams. Popular Agile methods, such as Scrum, tend to isolate teams from the business side and seem to have forgotten management’s role altogether. These are critical aspects of all successful organizations. Here are some key elements:

- Business provides the vision and direction; properly selecting, sizing and prioritizing those products and enhancements that will maximize your investment.
- Teams self-organize and do the work; consistently delivering value quickly while reducing the risk of developing what is not needed.
- Management bridges the two; providing the right environment for successful development by creating an organizational structure that removes impediments to the production of value. This increases productivity, lowers cost and improves quality.

BECOME A LEAN-AGILE ENTERPRISE

Involve all levels. All levels of your organization will experience impacts and require change management. We help prepare executive, mid-management and the front-line with the competencies required to successfully change the culture to a Lean-Agile enterprise.

Prioritization is only half the problem. Learn how to both prioritize and size your initiatives to enable your teams to implement them quickly.

Learn to come from business need not just system capability. There is a disconnect between the business side and development side in many organizations. Learn how BDSD can bridge this gap by providing the practices for managing the flow of work.

WHY NET OBJECTIVES

While many organizations are having success with Agile methods, many more are not. Much of this is due to organizations either starting in the wrong place, such as focusing on the team when that is not the main problem, or using the wrong method, such as using Scrum or kanban because they are popular.

Net Objectives is experienced in all of the Agile team methods (Scrum, XP, Kanban) and integrates business, management and teams. This lets us help you select the right method for you.
LEARN TO DRIVE DEVELOPMENT FROM THE DELIVERY OF BUSINESS VALUE

What really matters to any organization? The delivery of value to customers. Most development organizations, both large and small, are not organized to optimize the delivery of value. By focusing the system within which your people are working and by aligning your people by giving them clear visibility into the value they are creating, any development organization can deliver far more value, lower friction, and do it with fewer acts of self-destructive heroism on the part of the teams.

THE NET OBJECTIVES TRANSFORMATION MODEL

Our approach is to start where you are and then set out a roadmap to get you to where you want to be, with concrete actionable steps to make immediate progress at a rate your people and organization can absorb. We do this by guiding executive leadership, middle management, and the teams at the working surface. The coordination of all three is required to make change that will stick.

OUR EXPERTS

Net Objectives’ consultants are actually a team. Some are well known thought leaders. Most of them are authors. All of them are contributors to our approach.

SELECTED COURSES

Executive Leadership and Management
Lean-Agile Executive Briefing
Preparing Leadership for a Lean-Agile Transformation

Product Manager and Product Owner
Lean-Agile Product Management

Lean-Agile at the Team
Acceptance Test-Driven Development
Implementing Team Agility
Team Agility Coaching Certification
Lean-Agile Story Writing with Tests

DevOps
DevOps for Leaders and Managers

Train the Trainer
Online Coaching Academy
Becoming a Team Agility Trainer

Technical Agility
Advanced Software Design
Design Patterns Lab
Effective Object-Oriented Analysis and Design
Emergent Design
Foundations of Sustainable Design
Sustainable Test-Driven Development

SAFe®-Related
Leading SAFe® 4.0
Using ATDD/BDD in the Agile Release Train (workshop)
Architecting in a SAFe Environment
Implement the Built-in Quality of SAFe
Taking Agile at Scale to the Next Level

OUR BOOKS AND RESOURCES