The Project to Product Transformation

Practical Guidance from Fourteen Enterprise Journeys

IT REVOLUTION
DEVOPS ENTERPRISE FORUM
2019

25 NW 23rd Pl.,
Suite 6314
Portland, OR 97210

The Project to Product Transformation

Produced in the United States of America

Cover design and interior by Devon Smith

For further information about IT Revolution, these and other publications, special discounts for bulk book purchases, or for information on booking authors for an event, please visit our website at ITRevolution.com.

The 2019 DevOps Enterprise Forum was sponsored by XebiaLabs.

Preface

In May of this year, the fifth annual DevOps Enterprise Forum was held in Portland, Oregon. As always, industry leaders and experts came together to discuss the issues at the forefront of the DevOps Enterprise community and to put together guidance to help us overcome and move through those obstacles.

This year, the group took a deeper dive into issues we had just begun to unpack in previous years, providing step-by-step guidance on how to implement a move from project to product and how to make DevOps work in large-scale, cyber-physical systems, and even a more detailed look at conducting Dojos in any organization. We also approached cultural and process changes like breaking through old change-management processes and debunking the myth of the full-stack engineer. And of course, we dived into the continuing question around security in automated pipelines.

As always, this year's topics strive to address the issues, concerns, and obstacles that are the most relevant to modern IT organizations across all industries. Afterall, every organization is a digital organization.

This year's Forum papers (along with our archive of papers from years past) are an essential asset for any organization's library, fostering the continual learning that is essential to the success of a DevOps transformation and winning in the marketplace.

A special thanks goes to Jeff Gallimore, our co-host and partner and co-founder at Excella, for helping create a structure for the two days and the weeks that followed to help everyone stay focused and productive. Additional thanks goes to this year's Forum sponsor, XebiaLabs. And most importantly a huge thank you to this year's Forum participants, who contribute their valuable time and expertise and always go above and beyond to put together these resources for the entire community to share and learn from.

Please read, share, and learn, and you will help guide yourself and your organization to success!

—Gene Kim
June 2019
Portland, Oregon

Collaborators

Ross Clanton
Executive Director of Technology Transformation, Verizon

Amy Walters
Product & Agile Transformation, U.S. Bank

Jason Zubrick
CTO, defi SOLUTIONS

Pat Birkeland
Business Analytics Platform Services, Boeing

Mik Kersten
Founder and CEO, Tasktop

Alan Nance
Co-Founder, CitrusCollab and External Advisor, Royal Schiphol Group

Anders Wallgren
Vice President of Technology Strategy, CloudBees

Contents

Introduction

Over the last couple of years, the DevOps Enterprise Forum has written guidance on transforming businesses through the understanding and reconciling of value streams with papers like *Value Stream Architecture: Creating an Architecture to Connect the Dots in DevOps* and *Moving from Project to Product: Modernizing Traditional Enterprise Operating Models*. These papers provided guidance on product transformation from a theoretical perspective but lacked real-world advice to help practitioners apply the guidance. Instead of another paper on *why*, what we need now is more guidance on *how*.

This year, we felt it was important for us to put our money where our mouth is. We used the decades of experience from our authors and peers at the DevOps Enterprise Forum to glean as much as we could about the do's and don'ts of enterprise product transformation, and we compiled that knowledge into something digestible and applicable to you and your organizational transformation.

The target audience for this paper is leaders at any level of the enterprise who are driving said transformational change—more specifically, transitioning enterprise IT from a traditional project operating model to a product operating model. Transformations that shift from project to product require changes at every level and in every corner of the enterprise. If you are a leader trying to drive a product transformation in your organization, this paper is for you.

To gather information that is both relevant and that has been proven successful, we interviewed industry thought leaders across fourteen large enterprises who successfully drove product and technology transformations. These leaders had amazing insights from their successes and failures in moving organizations from project to product at scale. We took the enlightening moments from those interviews and broke the learnings and recommendations down into exercisable guidance. Through the interview process, we noticed similarities and themes that formed the main structure of this paper. The interviews conducted intentionally spanned a breadth of industries to demonstrate the applicability of the product operating model in any industry.

The companies assessed span the following industries: retail, banking, airline, telecommunications, apparel, accessories, sports equipment, financial services, insurance, technology, food and beverage, and medical software.

In the following sections, there is a distinct hierarchy of relationships within our guidance. The transformation is broken into domains as well as different phases or periods of the transformation, which we call stages. These segments contextualize the struggles and hurdles that occur at different times during the transformation and within each of those stages. We also took the learnings from our interviews and created straightforward guidance we call indicators.

Product Transformation Domains

When thinking about transformations of any kind, it can be difficult to visualize all the factors involved. By definition, transformation is “a thorough or dramatic change in form or appearance.” This is especially challenging when it comes to changing individual behaviors. As you can imagine, the difficulties in fighting the inertia of an enterprise to move from a project-based delivery model to a product-centric model can be defined with this generic definition of transformation: dramatic change. Dramatic and daunting.

To help leaders grasp the complexities and difficulties of implementing change within the enterprise, we have defined common areas of focus for categorizing those changes. These categories are referred to in this paper as *Product Transformation Domains*. These domains are the highest level of focus for the change and are designed to help focus your change.

Next, we will dive into each of the domains of a product transformation journey and give a simple definition to set the context for transformational change.

The Seven Domains

Figure 1 shows the seven domains of product transformation. Each domain contains unique properties that may stand alone or have interrelated dependencies to the other domains.

- **Transformation Implementation:** Provides guidance for getting the transformation started, driving it, and keeping it alive and well.
- **Business and Technology Synchronicity**: All successful transformations bring business and technology organizations closer together. This domain outlines how to tackle the problems of engagement and alignment across the organization. Additionally, it dives into approaches for effective planning and prioritization across technology and business in a product model.
- **Product Taxonomy:** Arguably the most difficult domain to explain because it can take on so many different forms from its inception. As you’ll read later in

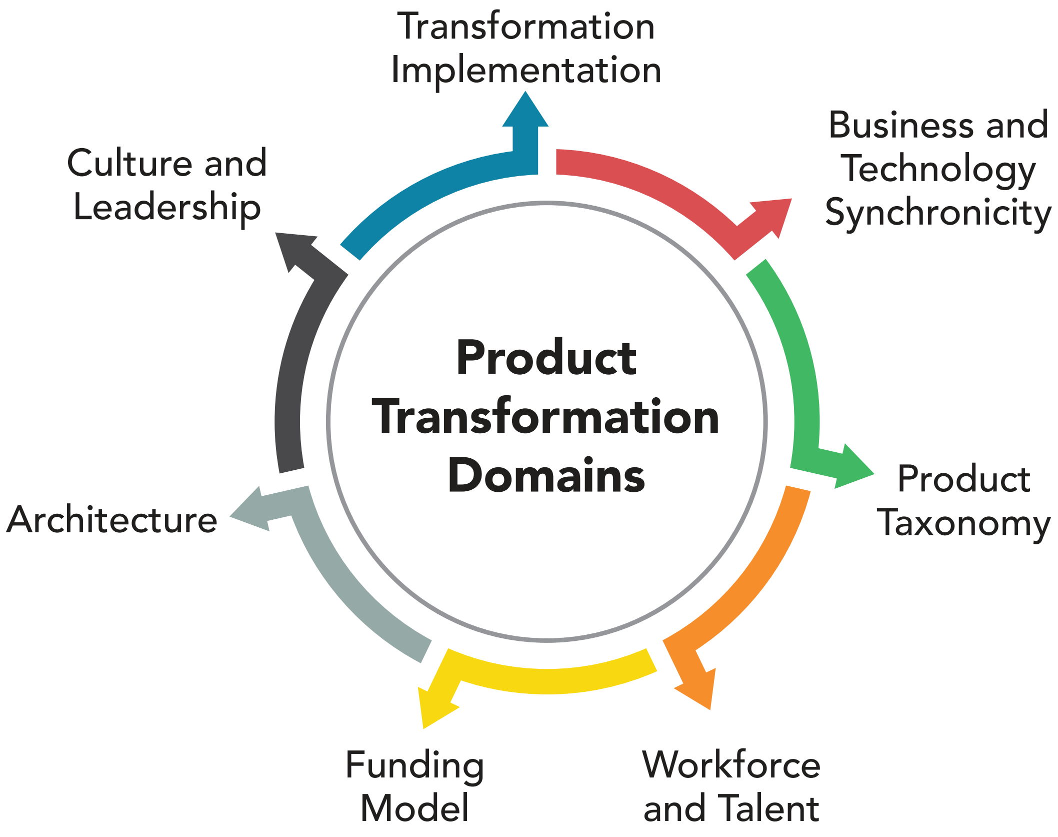

Transformation Implementation
Business and Technology Synchronicity
Product Taxonomy
Workforce and Talent
Funding Model
Architecture
Culture and Leadership
Product Transformation Domains

Figure 1: The Seven Product Transformation Domains

this paper, the Product Taxonomy domain is at the center of designing a product-centric organization.

- **Workforce and Talent:** Focuses on organizational size and structure, new team roles, and fixed versus variable labor. Additionally, this domain focuses on how to "level-up" employees as they move into new roles in the organization.
- **Funding Model:** Addresses how to change funding models to support a product-centric operating model. It is often the most difficult domain to change at the enterprise level because it crosses so many boundaries.
- **Architecture:** Allows the transformation to be implemented across a diverse set of tools, software, and infrastructure components used to deliver software in the organization. This domain also encompasses the concepts of value stream architecture and how organizations are taking progressive architectural decisions to deliver value faster.

- **Culture and Leadership:** Focuses on how to ignite your culture and engage leadership to ensure you have a successful transformation.

These seven product transformation domains contain numerous examples and learnings from those who have gone through this process and shared their learnings with us. In order to simplify the domains even further, we have added another facet to the guidance for your transformation process. From the data we collected, interviews we conducted, and our own experiences, we noticed three distinct stages that can significantly alter the strategy and tactics applied toward the implementation. The following section will define the three stages and how they relate to the seven product transformation domains.

The Three Stages of Product Transformation

A product transformational effort has multiple stages. Transformational efforts as invasive as these are organic and take on a life of their own. They require different approaches and guidance along their different stages of existence to stay healthy and maintain practicality to the enterprise.

Through our research and experience, we identified three distinct stages of life for any product transformation: *incubate*, *scale*, and *optimize*.

- **Incubate:** Incubation is the earliest stage of product transformations. This is the stage in which early adopters experiment and find other like-minded thought leaders to move the organization forward. Incubation is often found in pockets of the enterprise and may or may not apply to all domains in every organization. Incubation, as with the other two stages, can't be defined by a timeline. Stage timelines are unique to each organization, and just as every living thing has a unique existence, so too does each of the stages within a product transformation life cycle.
- **Scale:** Scale is the stage in which most of the transformational change happens. Think of scale as the stage when your transformation enters its teen years and then transitions through its twenties, thirties, and forties. In this stage, your product transformation will typically be driven across most of the

enterprise. Incubation built a great foundation for change, convinced the right people, and showed enough value to keep the transformation moving forward. Scale is where the ramp-up begins and the entire enterprise is mobilized and pushed to transform.

- **Optimize:** The optimization stage is where the enterprise takes what it learned over the scale stage and continuously improves. Often termed continuous or relentless improvement, the optimization stage continues refining the transformation through measured learnings and tighter business/technology synchronicity.

Recommended Indicators

During the process of researching and interviewing for this guidance, our team found commonalities in our data. The commonalities took many forms, like keywords in interviews to recognizable sub-themes in research and experiences. We then broke our findings down into a matrix of domains and stages. Where each domain met one of the stages, we extrapolated our findings to fit this domain/stage model. We are calling these findings *indicators*.

An interesting byproduct of our process was the constraint indicators we found and documented. Within each domain/stage convergence, we focused on the indicators where guidance would help to increase the velocity of transformation. All the transformation stories we analyzed had numerous “areas and actions to avoid,” and as a result, we realized a framework used to reduce friction was necessary.

Throughout the upcoming sections, you will find not only the guidance indicators to create, increase, and sustain velocity, but also the negative force learnings that should help you avoid pitfalls in your transformational journey. While this overall framework is shown in Figure 2, each of the domains will be explained in more detail in the subsequent sections.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Transformation Implementation	• Align case for the 'why.' • Find a strong business sponsor. • Help teams that want to change. • Build a coalition with key thought-leaders. • Establish clear outcome measures.	• Expand communication through town halls. • Ensure there is strong C-level buy-in. • Align business strategy with technology strategy. • Leverage internal leadership to drive transformation. • Leverage change-management function to drive communication.	• Shift Agile/product delivery to business teams. • Use Dojos to focus on business priorities. • Expand communication to the enterprise level.	• Technology transformation is disconnected from business strategy. • Lacking in strong C-level sponsorship. • New senior leaders that don't share these beliefs.
Business and Technology Synchronicity	• Initial changes typically more technology-centric. • Introduce the concept of Lean Value Streams. • Experimental teams leverage new prioritization model. • Align to overall business strategy.	• Parallel technology and business organizations around products. • Expanded focus on product-based outcomes. • Lightweight quarterly product planning (QPP). • Drive a data-driving organization. • LEAN prioritization process.	• Technology and business operate as one team. • Product owners are authoritative source for priorities. • Create self-organizing "product networks." • Establish routines around discovery.	• Excessive politics and organizational bureaucracy. • Product planning exercises with high overhead. • Fake Agile. • Partially allocated or not fully dedicated product team members.
Product Taxonomy	• Start with mapping value streams and/or customer experiences. • Leverage existing taxonomies (e.g., Business Capability Frameworks) as starting point. • Identify business capability owners. • Establish clear definition of "What is a Product?"	• Map organization to portfolios, product groups, products. • Establish product and platform teams, all following product management practices. • Structure teams around "you build it, you run it."	• Continuous improvement of taxonomy.	• Lacking clarity of definition and product mapping lead to excessive overlaps. • Blindly following an Agile framework without first thinking through your products.

Workforce and Talent	• Start with Agile and Dojo boot camps. • Ensure a strong Prod-uct Manager focus. • Define "Technical" and "Experience" based Product Managers. • Ensure a capacity-based model for work commitment.	• Focus on internal employee insourcing. • Reduce management layers and optimize team sizes. • Scale boot camps/ Dojos. • Reduce non-technical roles.	• Drive for very lean organizational structure. • Evolve Scrum Masters into coaching roles. • Break down/pivot PMO organization.	• Functional outsourcing. • Functional siloing across the technology organization. • PMO-driven transformation. • Providing inadequate training.
Funding Model	• Establish pilot teams as capacity-driven. • Work within budget and financial constraints. • Use a quarterly fund-ing model for piloting teams.	• Shift from project funding all teams for technology investments. • Release funding quar-terly based on value delivered. • Create a "Strategic Investment Committee." • Epic-based funding/"Shark Tank."	• Continue to focus on alignment on business priorities and product funding. • Expand funding model to non-technology investments.	• Technology organi-zations are treated as cost centers. • Business leaders expect to fund features.
Architecture	• Establish flexible, low-friction ability to build. • Introduce architectural patterns that help with speed of delivery. • Engage enterprise and business architecture organizations.	• Create "toolchain as a product" paradigm. • Focus on automation. • Adopt to open-source platforms and tools. • Create architectural communities.	• Federate architectural decisions for speed to market. • Move to cloud and cloud native.	• Changing operat-ing model without deliberate focus on decoupling systems. • Bureaucratic architecture processes impede team velocity.
Culture and Leadership	• Focus on establishing community. • Create a bottom-up cultural movement. • Establish the proper incentives to enable empowerment and cohesion.	• Build culture of con-tinuous learning. • Leverage internal DevOps Days and Dojos. • Use transformational leaders to establish relationship with "bottom-up" transfor-mational movement. • Focus on execu-tive immersion and learning. • Emphasize "fail-ing fast" and team empowerment.	• Become members and influencers in external community. • Leaders teach prac-tices to other leaders.	• Executive behaviors contradict cultural aspirations. • Not addressing resis-tance to change from middle management. • Existing talent not being on board, lead-ing to attrition. • Leaders resisting increased transpar-ency, wanting to keep IT as a black box.

Figure 2: Product Transformation Indicators

Transformation Implementation

Product Transformation Guidance—Transformation Implementation

Figure 3 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Transformation Implementation	• Align case for the 'why.' • Find a strong business sponsor. • Help teams that want to change. • Build a coalition with key thought-leaders. • Establish clear outcome measures.	• Expand communication through town halls. • Ensure there is strong C-level buy-in. • Align business strategy with technology strategy. • Leverage internal leadership to drive transformation. • Leverage change-management function to drive communication.	• Shift Agile/product delivery to business teams. • Use Dojos to focus on business priorities. • Expand communication to the enterprise level.	• Technology transformation is disconnected from business strategy. • Lacking in strong C-level sponsorship. • New senior leaders that don't share these beliefs.

Figure 3: Transformation Implementation Indicators

Why Is This Important?

Large-scale transformations in enterprises often fail because it can be difficult to successfully change the structure, ways of working, and culture across thousands of people. There are so many factors that can work against your transformation agenda, ranging from poor linkage between strategy and execution to resistance to change within the leadership team or from the teams themselves. Most importantly for success, people need to understand why the enterprise is transforming. They need to understand what problems will be solved (e.g., speed to market, customer obsession, digital disruption, IT spend, etc.).

Enterprises often take multiple years to work through a full-scale transformation from project to product. Having a clear vision as well as a sound strategy and approach for your transformation implementation is critical to increasing the likelihood of success.

What Is It?

Your transformation implementation strategy helps you to establish the goals, approach, and rollout for your transformation. The transformation can be approached in many ways: in phases, by lines of business, by certain portfolios that meet specific criteria, by experimentation followed by growth, or by a "big bang" approach.

In the next sections, we will dive in further to understand how enterprises are aligning technology and business strategy as part of their transformation implementation. There are many different pathways an organization can take to achieve a full-scale project to product transformation. We intend to provide a set of patterns that leaders driving transformation can utilize.

Our proposed strategy template suggests how to handle communication to executives, middle management, and to the broader organization, as well as how to handle sponsorship, change management, and governance for these changes. And equally as important, we define what measures should be used for success and how those measurements get established across the organization.

We will also explore the size, scope, and complexity of different enterprise transformations and how the implementation strategy may have been positioned differently based on different scenarios. While these transformation tactics can prove useful in many types of companies, this paper is primarily focused on the experiences of large enterprises that are part of the *Fortune* 500.

How Are Enterprises Achieving Success?

The fourteen enterprises we assessed varied in size, from a few thousand to well over twenty thousand people. The assessments include entire technology organizations as well as the people responsible for product ownership. Typically positioned within the business side of organizations, product ownership, in rare cases, was also found in the technology organization. Product owners were primarily impacted in that their planning/prioritization approach, team structures, workflow, and sometimes even their individual roles changed. Changes to those respective areas will be discussed in more depth in the subsequent domains.

Start Small, Then Spread

A general theme across enterprises is that transformations typically start small in order to demonstrate results and build momentum, often before they even formally align an executive sponsor. A common practice is to find a few teams who want to change and help them. At this incubation stage of the transformation, a small group of progressive thought leaders will start framing a strategy and approach together. Then, by helping the small group of early-adopter teams shift to a product-based operating model, these leaders are able to leverage the small success to inform and inspire a broader transformation strategy. Ultimately, to scale the transformation, you need to secure a strong transformative executive sponsor, or even multiple sponsors, ideally in both the business and technology organizations. You'll need to leverage these sponsors to secure strong C-level buy-in for the transformation.

After incubating the transformation in a few, small early-adopter teams, enterprises must decide how to best roll out the change across the broader organization. In most cases, enterprises chose to implement the change by line of business (LOB) or by portfolio. The most common path we saw was to start the transformation in more digital-focused and customer-facing areas. There is often more pressure for these areas to move to a product-centric model more quickly, as companies are increasingly competing based on the speed at which they can delight their customers with new experiences and features.

While it can be a good practice to start with the more external customer-facing products, it is not wise to stop there, as Gartner's theory of bimodal IT—the practice of managing two separate but coherent styles of work, one focused on predictability and the other on experimentation, at once[1]—might suggest. As these customer-facing teams start to transform, pressure is placed on the broader organization. You now have different parts of your organization using different processes for planning, prioritization, and delivery. Team structures and roles are different. Tooling will likely even be different. This creates many inefficiencies, especially as the old model and the transformed model mash against each other when something needs to be delivered across multiple parts of the organization. Additionally, it also starts to create a culture of the "haves" and "have-nots," which can negatively affect overall morale of the organization.

Most of the companies we interviewed continued to expand their transformation beyond these initial portfolios and eventually transformed all their technology portfolios, including infrastructure. By expanding the transformation across more of the enterprise, and eventually across the whole enterprise, the pressure created by two different models can be alleviated.

Communication is Key

The different enterprises we assessed had various tactics for how they approached the overall transformation communication strategy, both in how they introduced the change as well as their ongoing communication approach. Generally, within the incubation phase, there wasn't a heavy focus on communication, as these organizations were more focused on experimenting, learning, and attaining initial results that they could build from. However, as they looked to make bigger changes across the organization and further grow the transformation, the communication strategy and approach became extremely important. Some of the most effective techniques are as follows:

- **Frame the "why"**: Beyond communicating what the operating model changes were going to be, how they were going to be implemented, and when, the enterprises focused on framing *why* they were changing in the first place. They made it clear why the changes were critical to the long-term success of their company. Typically, they tied the why back to disruptive factors in their industry, as well as to the increasing demands of speed and agility from their customers and business overall.
- **Initiate internal change management:** Internal leaders were responsible for this activity, participating in the strategy and planning for the transformation. These leaders focused on how to pull out key messaging to help frame their transformations to different stakeholder groups within the organization. They relentlessly looked for powerful sound bites that could be captured through

these discussions. Additionally, they managed an overall communication plan and ensured that there was a constant and consistent flow of information through the organization.

- **Focus on transparency:** For many, transparency was achieved through continual leadership town halls to share plans and messaging as they unfolded. Business and technology executives met with teams during these town halls to have informal conversations about the transformation. Some companies even mixed executives up from different parts of the organization. The goal of these town halls was to show that the executives were all working together toward the same outcomes. These town halls were completely open to any questions the audience was concerned about, and they were surprisingly beneficial in several ways. First, the executives showed vulnerability by fielding questions that they couldn't necessarily answer. Their openness and vulnerability helped build trust across the organization. Additionally, by making the company's transformation a conversation that everyone had an opportunity to participate in, everyone had a voice. Individuals not directly involved in guiding the transformation were able to share their ideas and concerns, which helped them feel they had input into the transformation plans.
- **Use success stories:** After helping early-adopter teams, their successes were leveraged by encouraging those teams to go out and share their stories. This works best when business leaders and technology leaders are both telling the story. Use these success stories to build more excitement and energy for what is to come. These stories can help engage the rest of your management team, especially if there is any competitive aspect of your culture.

Track and Measure Success

The measures used to track and govern the transformation across the organization typically focused on a few key areas. About half of the measures focused on the process of delivering technology whereas the other half focused on the value and outcomes that were delivered. In measuring technology delivery, most companies were anchored in either the measures defined by Dr. Nicole Forsgren, Jez Humble, and Gene Kim through the *State of DevOps Report* and their book *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*, or the Flow

Framework™ measures as defined by Mik Kersten in his book *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*.

A common initial key measurement organizations used was cycle time or end-to-end lead time, which an enterprise would use to standardize each product's timeline. Even though cycle time isn't truly an end-to-end measure, it does typically cover a product's life cycle from the point a story is pulled out of a backlog through the product's implementation. Over time, organizations have shifted their product timelines further left to when the business idea is originally conceived.

Outcome measures used for the transformation tended to focus on business outcomes, such as revenue generation, customer acquisition, conversion, cost reduction, and/or customer/employee Net Promoter Scores (NPS). Many enterprises also shifted to an objectives and key results (OKRs) method. This allowed them to set up their governance model around tracking these OKRs, which have further connections into how leaders planned, prioritized, and funded the transformation, which will be discussed in more detail later.

Finally, from a transformation implementation perspective, some enterprises leveraged outside help from management consultants. Experiences with these consultants were mixed. Often, consultants were chosen by senior executives to work with the internal leaders of the transformation to frame the strategy and drive the change-management and governance agenda across the organization. However, the companies that seemed most successful had very active *internal leaders* driving the transformation across the organization. Don't delegate the role to drive the change to outside consultants. We believe that without passionate leaders inside the organization willing to lead and champion this charge, your likelihood of success will decrease.

Constraints That Need to Be Overcome

In our interviews, clear patterns emerged regarding constraints leaders needed to overcome during the transformation. In some cases, these constraints set back or even derailed the product transformation.

- **Technology transformation disconnected from business strategy:** Many companies started their transformation models in their technology. In doing so, they were holding off on tackling the complicated problem of aligning the business executives with the new model. This is fine early on when you are experimenting. However, when you are looking to scale your transformation and tackle bigger foundational elements in your operating model, such as prioritization processes, funding procedures, and roles, you will not be successful if the business isn't engaged. And, you will not be able to engage your business leaders if you can't frame your technology transformation in the context of how it helps them achieve their business strategy. Spend time early on to understand your business strategy, what challenges they have, and how transforming your technology capability is going to accelerate your business' overall agility.

- **Transformation lacking strong C-level sponsorship:** A product transformation is properly implemented without a high level of commitment and top-down support. If you don't have C-level leaders modeling the way a product-centric enterprise should function and they are not personally leading the charge in the transformation, the resistors within your organization will likely undermine your change, both actively and passively. Bottom-up momentum will only go so far. If you focus on framing the technology transformation in a way that drives your business strategy and appeals to stakeholders, it will help secure this support.
- **Changing senior leadership:** We learned of a few cases where a new senior leader joined the company from an organization that had differing cultures and beliefs. If these new leaders were not educated or experienced in the value of Lean, Agile, DevOps, and product-oriented practices, they tended to rely on the practices and behaviors that had helped them be successful previously. It is startling how fast these new senior executives can unwind years of transformation progress. This constraint is somewhat linked to the previous ones. If there is strong C-level sponsorship and you've built more support across business and technology executives, then you are less likely to hire new executives who don't align with this direction, and even if they are hired, you will have more support to counteract any regressions they may try to make.

Business and Technology Synchronicity

Product Transformation Guidance—Business/Technology Synchronicity

Figure 4 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Business and Technology Synchronicity	• Initial changes typically more technology-centric. • Introduce the concept of Lean Value Streams. • Experimental teams leverage new prioritization model. • Align to overall business strategy.	• Parallel technology and business organizations around products. • Expanded focus on product-based outcomes. • Lightweight quarterly product planning (QPP). • Drive a data-driving organization. • LEAN prioritization process.	• Technology and business operate as one team. • Product owners are authoritative source for priorities. • Create self-organizing "product networks." • Establish routines around discovery.	• Excessive politics and organizational bureaucracy. • Product planning exercises with high overhead. • Fake Agile. • Partially allocated or not fully dedicated product team members.

Figure 4: Business and Technology Synchronicity Indicators

Why Is This Important?

So, how can enterprises establish a product management capability with product managers who understand business/technology alignment? The evolution of project-based delivery to product-centric value streams intends to create a frictionless flow from business architecture through product development to valuable outcomes. This fundamental change in an organization's workflow focuses on the orchestration of integrated value creation as opposed to a relay race of handoffs toward value.

A significant outcome that enterprises are trying to achieve by moving to a product-centric model is to prioritize their customer. Focusing on products rather than projects enables enterprises to better orient their teams around customer outcomes. Further, by basing the engineering organizations within IT on the same model, the product can be seamlessly exchanged between parallel engineering and product organization teams. This type of structure allows enterprises to optimize for speed and agility by significantly reducing the waste and handoffs that typically exist in traditional operating models.

What Is It?

The business and technology synchronicity domain addresses the enterprise's structured and the engagement models needed across technology and business teams during the shift to a product-centric operating model. This is the structural change that makes business digital transformation possible. (The primary organizing construct that companies typically follow to drive their product alignment will be discussed later in this paper.)

Not only do we need to look at the structure of the teams, but also how work is organized, prioritized, and delivered across the organization. It is not simply a *software development* life cycle but a *product* life cycle. We will explore the roles between the business product organization and its IT counterpart as they relate to intake, prioritize, and the delivery of the product. A key outcome of this domain is to create an environment in which business and technology leadership fully engage in product development together.

How Are Enterprises Achieving Success?

Pivot from Project to Product

At the core of the change to the operating model is the idea that organizations need to pivot from a model where they optimize for cost and efficiency to one where they optimize for speed and agility. Traditionally, since most enterprises optimized for cost and efficiency, they created disparate teams centered on individual aspects of the production cycle, such as analysis, development, operations, and quality assurance. The philosophy around the traditional model was that each team could optimize their efficiency. Ironically, scaling this model across enterprises led to significant inefficiencies, not because each team wasn't efficient with the localized process for their function, but because it was highly inefficient to manage the work end to end across the entire software delivery value stream. As a result, waste built up in the technology delivery processes and, at an aggregate level, the costs of technology increased.

Leading enterprises have completely flipped this traditional model from one where teams are organized around functions and outputs to a model where teams are now organized cross-functionally around customer outcomes. Introducing this change into an enterprise is extremely difficult. For many companies that we interviewed, these changes largely started in their technology organizations, as they were treating these transformations as a technology strategy. However, the organizations that were able to deliberately tie this transformation to their overall business strategy and treat it as a joint business and technology transformation saw the most success. Additionally, leading enterprises were more data-focused and made their progress and challenges visible and transparent. This helped align business and technology, as previously technology delivery was often viewed as a "black box" by their business counterparts.

Prioritization

Prioritization changes significantly in a product-centric operating model. The project-based model typically results in businesses defining their requirements up front, often at the beginning of the year before the project is even funded. Then the engineering organizations look at all the requirements for the entire project and estimate the level of effort needed to fulfill them. The estimation then sets what the funding needs are, and the project gets funded if it appears to have a positive ROI (return on investment).

There are many well-documented concerns with this process. Many of the requirements aren't typically needed, it is nearly impossible to accurately estimate what it will take to deliver them, and the customer is missing from this equation entirely. In fact, the first time the customer gets exposed to what is being delivered is typically many months later (or even years), which is a significant anti-pattern in a product-centric model.

Even though this isn't an ideal method of prioritization, it is relatively entrenched in most enterprises. When in the incubation stage, most of the early teams try to operate within the existing model. This means they set up a separate prioritization model for their team, but they must do extra work to integrate it back into the model dictated by the Project Management Office (PMO). While this isn't ideal, this technique is often leveraged by teams just getting started.

As enterprises hit their scaling stage, they begin to introduce leaner prioritization models where they pull key product stakeholders together on a relatively frequent cadence to discuss priorities and dependencies. Then, the respective leaders for the different products commit to delivery for the upcoming iterations. This is a huge improvement from the use of a single session of planning and prioritization at the beginning of projects.

From a strategic planning perspective, leading enterprises typically move to a quarterly product planning model (or twice annually). This is not the heavy-weight program increment planning that many of the companies who have implemented Scaled Agile Framework encounter, but a lighter-weight model that is focused on getting product owners to commit to their quarterly OKRs, which should tie to the funding model (which will be discussed in the next section).

As these enterprises further advance into an optimized state, the quarterly product-planning routine evolves into self-organizing product networks with other product teams that have interdependencies. This exercise becomes a joint planning session to work through interdependencies and conflicting priorities.

Network-Based Organization

In fact, one key sign of success in optimized organizations is that they have pivoted from a hierarchical organization to a network-based organization of small teams who are able to collaborate horizontally without layers of management telling everyone what to do.

The product owners are empowered to connect with other product owners to align their product objectives with their value stream priorities and/or dependencies. The product owner teams then determine the level of frequency that they should connect to synchronize product roadmaps and OKRs, manage dependencies, and remove blockers for their respective teams. Product owners also start to become a more authoritative source on priorities instead of just a proxy representing many different business stakeholder interests.

With this shift, organizations take on a “you build it, you run it” philosophy. This can be incredibly difficult as there are typically leaders for these different functions, and this progressive model defies what many believe is conventional wisdom for operating a technology organization.

In the incubation stage, teams should be structured around the product organically. The teams aren’t yet at a point where they need a more formalized organizing construct. To ensure the highest possibility of success, these early teams need to be organized full-stack and full-life cycle so the organization can apply the product-based model on a small scale before pivoting the whole enterprise. We will describe these full-stack team structures further in the workforce and talent domain.

In these early stages, teams are often matrixed into a model where they still have formal reporting relationships following the traditional functional model. These early teams may still be relatively technology-centric as well, especially if the transformation started within the technology organization. In these cases, the product owner may come from IT. Generally, though, business management should be involved in this change as well.

Business and Technology Work as One

As the transformation begins to grow and scale across the organization, it is imperative to have business and technology working together. At this stage, begin to reorganize your teams around customer-centric product outcomes. You will need to align an organizing construct, such as business capabilities, which we will discuss further in the product taxonomy domain.

Most companies start to align their technology organization to how the business product organizations are structured. The business and technology executives work closely to align how they are leading their respective organizations. It is also an advantage when colocation of people is feasible. When teams sit together it can create a similar experience for different business and product groups. In specific cases we found that when most people were colocated geographically within a single business, complexity was limited.

For the most optimized enterprises, the product and engineering teams start to operate as though they are one organization. Product decisions happen at the team level. In some cases these teams physically meet to discuss their processes. In one example, the CTO of an organization we interviewed had reorganized all of the product management competency into the technology organization. At a minimum, aligning the product teams on a common set of OKRs can be a key tactic to get them working as one team.

Shift Discovery and Delivery Models from Outputs to Outcomes

As companies go through this transformation, most introduce the concept of Lean value streams into their organization. Value stream mapping provides visibility into the overall product ecosystem. This also starts to orient teams to focus on the delivery of outcomes, and it provides a great structure to measure and continuously improve your delivery process to achieve those outcomes. For many organizations we analyzed, this shift led to the formalized role of a value stream architect.

DevOps Practices in Place

Nearly all the organizations we interviewed were already somewhat mature in their DevOps practices. This is a critical capability that gives organizations the ability to scale into the product-based model. However, when most organizations start, they don't yet have control over the deployment of their applications, as that function is typically segmented to a separate team responsible for deployment. During the incubation phase, this responsibility must be given to the early experimental teams. It is critical to address this disjunction before scaling this model.

Incorporate Product Discovery

As organizations move to the optimized stage, they expand from focusing on delivery to also incorporating product discovery into their practices. This level of maturity is accomplished when transformations adopt a "test and learn" attitude as every advancement to the product is treated as an experiment. To achieve this, you must build product discovery capabilities into your organization as early in your transformation as possible. This experimental and iterative approach allows product owners to quickly determine whether the company should continue investing more time building out a set of features, or whether you should pivot to a different experiment. Transformed organizations establish routines around discovery and delivery.

Constraints That Need to Be Overcome

There are many barriers to changing the operating model found in organizations. The business and technology synchronicity domain requires a significant focus on changing management structure to achieve the desired transformation results. Some of the primary constraints are as follows:

- **Excessive politics and organizational bureaucracy:** Organizations create and use many processes over time. In fact, teams build process upon process, optimizing not for the end-to-end experience but for the team who built it. These processes become entrenched in the organization, and it can be hard to get people to pivot and re-evaluate them. Compliance is commonly used as the reason why these processes need to exist. Typically, there are many ways to solve the problems that these processes were created to address. In many cases, there are more engineering-centric ways to solve compliance problems that provide better control and efficiency. To get past this constraint, arm yourself with examples of how others (preferably in your industry) have simplified similar processes in their organizations. Suggest trialing a new approach on an extremely small scale. Call it an experiment and use the results to get support and momentum from others who are also frustrated with legacy processes. You will need to build support and momentum to convince most of these teams to change.
- **Product planning exercises with high overhead:** Many Agile and product-centric transformations fail due to the excessive bureaucracy that becomes layered into the model. This often happens when program management philosophy is the main force driving the transformation. We found little evidence that large groups of people need to come together for every program increment planning cycle. You shouldn't have excessive coordination between products. When operating in a highly coordinated release process, there is little incentive to modernize architecture and eliminate dependence on others. This is an anti-pattern that is commonly implemented in enterprises. The proper way to solve this is to do lightweight product planning. Product leaders should be able to connect with their interdependent products to align

priorities. Establish expectations and priorities for teams to reduce dependencies in the overall architecture.

- **Fake Agile:** Many Agile transformations fall short of their goals because they aren't tackling the hard changes necessary at the organizational and team levels to truly become Agile. They tend to lack focus on the Agile mindset and values. In these cases, enterprises will often adopt Agile frameworks, practices, and ceremonies without fixing their broken and inefficient model. Additionally, many of these unsuccessful transformations tend to focus on technology delivery while ignoring the important aspects of the value stream related to how business and technology teams interact with each other from a discovery and planning perspective. The focus on the customer is noticeably absent in these fake Agile transformations, with many handoffs and layers of bureaucracy in place between the idea and the customer.
- **Partially allocated or not fully dedicated product team members (aka "resources"):** Persistent or long-lived cross-functional teams who are fully dedicated to their product are critical to success. When shifting from a project-based operating model, it can be difficult to shift mindsets away from project-based resource allocation. Partial allocation of a team member leads to constant context switching, which is detrimental to the success and overall morale of a product team. Another critical cultural component to a healthy transformation is to stop referring to people as "resources." People are not resources or machines; they are experienced knowledge workers with skills and qualities that are critical to the success of a product.

Product Taxonomy

Product Transformation Guidance—Product Taxonomy

Figure 5 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Product Taxonomy	• Start with mapping value streams and/or customer experiences. • Leverage existing taxonomies (e.g., Business Capability Frameworks) as starting point. • Identify business capability owners. • Establish clear definition of "What is a Product?"	• Map organization to portfolios, product groups, products. • Establish product and platform teams, all following product management practices. • Structure teams around "you build it, you run it."	• Continuous improvement of taxonomy.	• Lacking clarity of definition and product mapping lead to excessive overlaps. • Blindly following an Agile framework without first thinking through your products.

Figure 5: Product Taxonomy Indicators

Why Is This Important?

Shifting to a product operating model starts with defining the products, which includes both internally facing and externally facing products. Defining the products is essential to identifying clear lines of ownership and accountability, and redesigning the organizational structure to align with customer needs instead of settling for traditional organizations that align around IT functions. When looking across the fourteen companies interviewed, it became clear that this domain is the most ambiguous and lacks industry guidance on how to define product taxonomy. The other pattern observed is that this level of product definition typically occurs during the scale stage and then is continuously refined based on key learnings throughout the optimize stage. Below, we provide emerging best practices that have been successful with companies that have progressed through the scale and optimize stages of product transformation.

A product taxonomy provides enterprises a visible, centralized method for defining, managing, and tracking products across the company's value streams. There are multiple benefits to defining the product taxonomy:

- Helps provide product visibility across business areas, revealing redundancies, dependencies, and value flow.
- Provides a common language for products within an enterprise, minimizing confusion between technology and the business.
- Provides clarity around product types, stages, whether the product is legacy or new, and whether it is strategic.
- Sets the stage for accountability and clear ownership of products. This creates a clear path for new features and support incidents. A product owner will continuously communicate product vision, strategy, and outcomes-based goals, synchronizing with other product leaders and ultimately driving toward enterprise goals.
- Creates awareness of workforce needs when the scope of products is unbalanced with the availability of product managers/owners from the business and engineering talent.
- Enables an enterprise to organize around customer needs instead of technology, applications, or IT functions.

What Is a Product Taxonomy?

To transition to a product operating model, you should first define your product taxonomy. The product taxonomy identifies the portfolio of products that exist in a given business area. The taxonomy helps to show the hierarchy and relationships between the different products and capabilities that exist within your business. Given their scale and complexity, enterprises often need a visible, centralized method for defining, managing, and tracking products across the company's value streams.

To fully understand how to map a product taxonomy, we must first anchor to a few basic definitions for what a product is and how to classify different product types. To define it simply for our context, a product is a software offering that we create to deliver value to our customer. By this definition it is implied that we know who our customers are; we understand what they value; we understand the ecosystem or value stream within which the product fits; and we design, build, and support our product as a cross-functional, long-lived team with the skills needed to own the product end to end. Meaning, if we design and build it, then we must also maintain and support it throughout the life of the product (in other words, "you build it, you run it"). Products can serve external customers and/or internal enterprise customers. Products can be differentiated by designating a *product classification* and *product life cycle stage*. Unlike projects, products do not have a predefined end date; instead, a product moves through life cycle stages. The life cycle stage and product classification definitions provided within this paper are suggestions intended to provide guidance. Figure 6 shows the product life cycle stages:

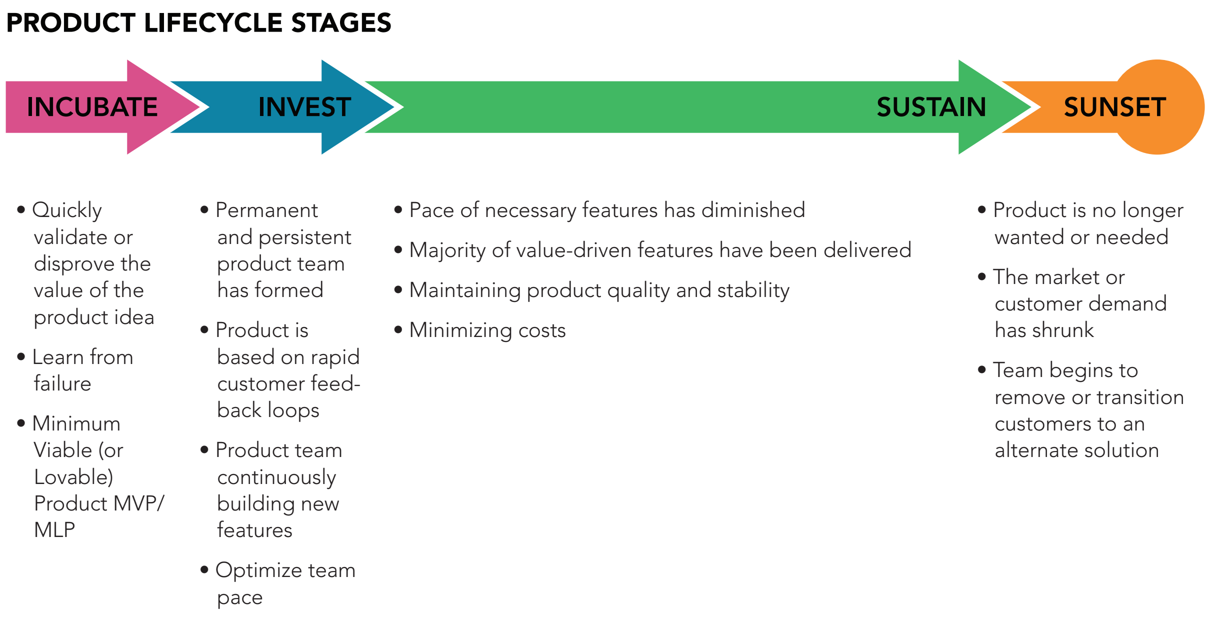

PRODUCT LIFECYCLE STAGES
INCUBATE
INVEST
SUSTAIN
SUNSET
• Quickly validate or disprove the value of the product idea
• Learn from failure
• Minimum Viable (or Lovable) Product MVP/MLP
• Permanent and persistent product team has formed
• Product is based on rapid customer feed-back loops
• Product team continuously building new features
• Optimize team pace
• Pace of necessary features has diminished
• Majority of value-driven features have been delivered
• Maintaining product quality and stability
• Minimizing costs
• Product is no longer wanted or needed
• The market or customer demand has shrunk
• Team begins to remove or transition customers to an alternate solution

Figure 6: Product Life Cycle Stages

Incubate

The product is a new idea or hypothesis that is rapidly being tested through product discovery, prototyping, and potential deployment of a minimum viable (or lovable) product (MVP or MLP). The goal of this stage is to quickly validate or disprove the value of the product idea. Exploratory product discovery may include a potential new business idea or model, a new technology set, or a customer opportunity space. Learning from failure is typical in this stage.

Invest

The value proposition of a product has been validated via a deployed MVP. A permanent persistent product team has been formed containing all the necessary skill sets to continuously design, build, and support the product based on rapid customer feedback loops. Product outcomes are measured regularly (e.g., quarterly) via clearly defined objectives and key results (OKRs). The product team continuously builds new features, proactively solves customer problems, and supports their product in production. Team pace, product use, and value are optimized based on customer and employee satisfaction, product stability and quality, and speed and adaptability.

Sustain

The pace of necessary new features has tapered off. The majority of the required functionality for the product has been accounted for. The team is primarily maintaining product quality and stability while minimizing maintenance costs.

Sunset

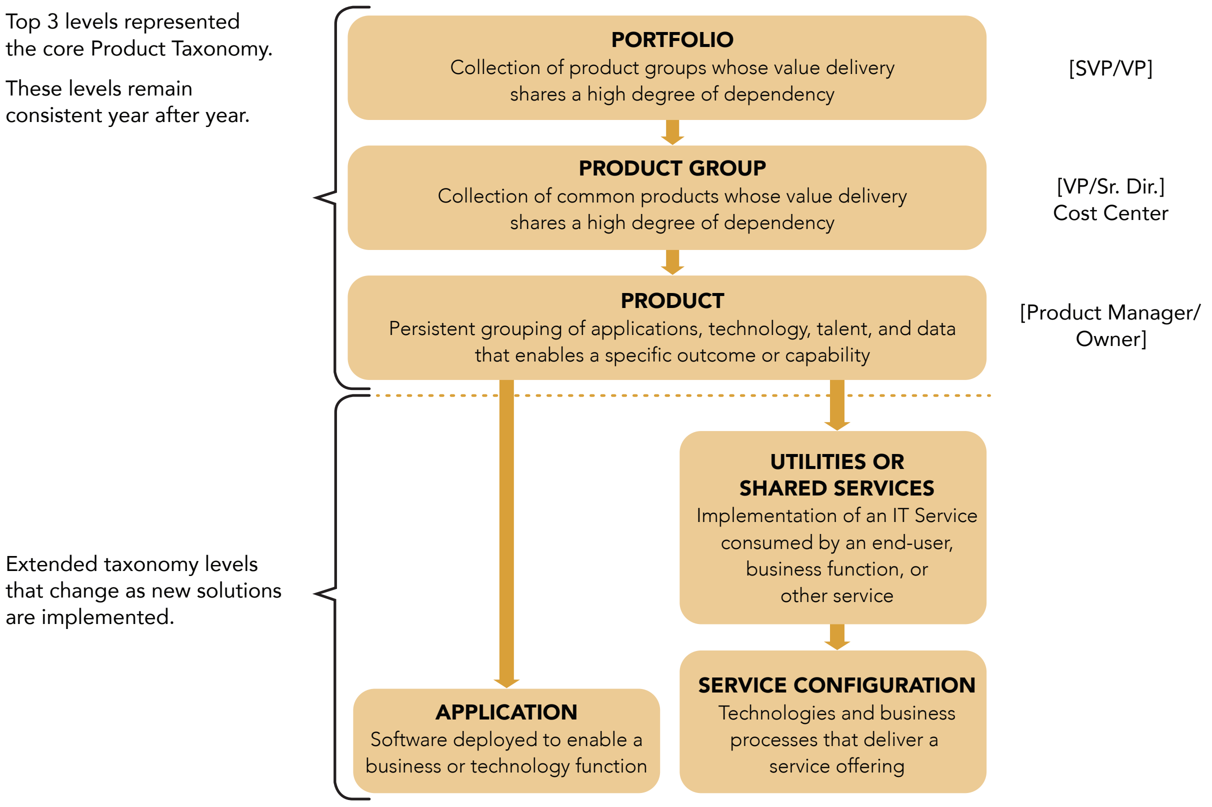
The product is no longer needed or wanted and the market or customer demand for the product begins to shrink; therefore, the product begins an end-of-product life stage. This is often referred to as the decline or sunset stage. The team works to remove or transition users to an alternate solution while decommissioning the product.

The Product Taxonomy Structure

The structure of a product taxonomy considers three levels or groupings of products. The first level is the product portfolio, which is a collection of product groups whose value deliveries share a high degree of dependency. The second level is the product group, which is a collection of related products whose value deliveries share a high degree of dependency. Lastly, the products are persistent groupings of applications, technology, talent, and data that enable a specific outcome, customer experience, or capability. When defining the product, focus on the outcome or customer experience it delivers rather than the application or technology that delivers the outcome or experience. Consider why a customer turns to your product or service and what problems it solves for them and evaluate that experience as the product. The technology or application is merely a vehicle for delivering that customer experience. The product should remain consistent year after year, whereas the supporting technologies will change over time as new solutions emerge. See Figure 7 for a representation of the product taxonomy structure.

PRODUCT TAXONOMY STRUCTURE

This is a suggested product taxonomy structure based on best practices observed through a couple of the leading organizations we'd interviewed.



Top 3 levels represented the core Product Taxonomy.
These levels remain consistent year after year.
Extended taxonomy levels that change as new solutions are implemented.
PORTFOLIO
Collection of product groups whose value delivery shares a high degree of dependency
[SVP/VP]
PRODUCT GROUP
Collection of common products whose value delivery shares a high degree of dependency
[VP/Sr. Dir.] Cost Center
PRODUCT
Persistent grouping of applications, technology, talent, and data that enables a specific outcome or capability
[Product Manager/ Owner]
UTILITIES OR SHARED SERVICES
Implementation of an IT Service consumed by an end-user, business function, or other service
APPLICATION
Software deployed to enable a business or technology function
SERVICE CONFIGURATION
Technologies and business processes that deliver a service offering

Figure 7: Product Taxonomy Structure

Product Types

Product types should be considered to help further differentiate your products. Does your technology product serve an external customer? Does your product enable other technology products within the enterprise? Is your product a shared service consumed by an end-user, business function, or another service? Ultimately, everything can be considered a product leveraging modern product management practices delivered by a long-lived product team. As we assessed the various companies we interviewed, the topic of inward-facing versus outward-facing products often came up. Inward-facing products were referenced as "IT products for IT" and outward-facing products were considered business products. Table 1 shows a few examples of product types with corresponding goals and examples.

Product Type	Goal	Examples
Customer-facing or customer experience	• Achieve product market fit • Revenue generating • Serves a customer need	Products a customer would purchase/ consume/use
Corporate- or vendor-facing business services	• Provide high-level capabilities, team member productivity, and operational agility	Payroll, virtual collaboration, vendor/ corporate interface, HR, general ledger, field support
Platform product or shared service	• Shared services, stability, scale, reuse, adoption, IT products for IT	Infrastructure, security, enterprise APIs, data systems, emerging technologies, shared service capabilities

Table 1: Product Types, Goals, and Examples

How Are Enterprises Achieving Success?

The companies we assessed appeared to struggle most with the product taxonomy domain; this domain appears to cause the greatest gap in transformation success due to the lack of documented references on how to map an enterprise product taxonomy. The natural starting point for most companies we interviewed was to anchor to business capability frameworks. Others leveraged their configuration management database, service catalogs, value stream maps, experience maps, or some combination of these tools. This section intends to map out how to work through this process.

Design a Product Taxonomy

The following are emerging best practices for designing a product taxonomy. This guidance is based on the experience from multiple companies in differing industries that have had successful transformations. Of the companies interviewed and researched, all companies in which we identified these practices were either in the scale or optimize stages of the product transformations.

These tactics are intended to provide guidance and to suggest approaches, tools, and techniques to help construct conversations that move the organization to a product-centric operating model. These steps are not a precise framework, nor are they linear or all-inclusive; rather, there are many ways to have these conversations. Appreciate that every company is at a different point in this transformational journey toward being more customer-obsessed, Agile-minded, and product-centric.

		Who	**Leading Questions & Considerations**	**Suggested Tools & Tactics**
Define the North Star for the Portfolio	**1**	IT & Business Leaders for a Portfolio	• What does success look like for our portfolio of products 1 year from now? • Where do we want to go? • Forward looking • Inspirational • Reflective of core values and culture • Aimed at improvements to the org in the future • Defines purposes	• Product charter • OKRs • Road map • Community map • Personas
Map Your Taxonomy of Products	**2**	Leaders (IT & BL)	• Why does your ream/space exist? • What is your north star or vision? • What capability or customer experience are you enabling/delivering? • What is your value stream(s)? • Are IT and the business aligned?	• Mapping end-to-end flow (value stream) • Mapping customer experiences • Map people, process, data, and technologies that enable value flow
Map Your Technologies to the Products	**3**	Leaders (IT & BL), Architect, Engineer	• What are the apps that are delivering/facilitating that experience? • Which apps should you own vs. not? Who should own them? Should they sunset? • What data do you need to support your product?	

Map Your People to the Products	**4**	Leaders (IT & BL)	• What skills and roles do you have? Are there gaps? • Is the product manager/owner identified? Do they come from the business or IT? • Beware of proxy product owners. • Is there an opportunity to automate manual tasks (e.g., testing)?	• Form full-stack product teams • Skills matrix • Product team roles • Identify blockers/risks
Identify Any Gaps You May Have (Skills, Teams, Roles, etc.)	**5**	Leaders & Teams	• Is your product team size 7 +/-2? Do you need to hire or repurpose existing roles? • Do you have employed engineers or contractors? What is the ratio? • Have you adopted DevOps? • Do you have traditional project roles or product roles?	
Create a Leader Action Plan	**6**	Leaders & Teams	• What blockers, risks, and constraints do you have to work through to move your org toward a product operating model? • Who can help? • Establish an action plan and work the items through to completion. • Keep blockers and actions transparent to all involved.	• Build a transformation backlog/kankan board
Coach, Charter, & Accelerate the New Teams	**7**	Leaders & Teams	• Once the teams are formed, launch then via coach supported chartering workshops to set their Agile/product/DevOps practices and working agreements. • Establish team collaboration space (co-locations).	• Engage a coach • Team working agreement • Team charter
Rationalize Products across Portfolios and Value Streams. Assess Redundancies and dependencies. GO! Expect an imperfect path. Pick a starting point and go! Learn. Iterate.				

Figure 8: Suggested Approaches and Tactics

Tactic 1: Define the North Star for the Portfolio

The north star is a shared purpose and vision that helps leaders and teams feel personally and emotionally connected. When we face adversity, this is our motivation to persevere. This is also a reference to validate that the activities we are executing are aligned with where we want to go.

Who Should Be Involved?

Technology and business leaders for the portfolio. It's recommended that technology and business leaders meet and align their goals at the beginning stages of mapping a product-centric operating model. Establish alignment and shared understanding across the leaders and then with the teams. Gather feedback from the teams and adjust as needed. What resonates with them? What doesn't? Make the north star relatable using clear, friendly language; steer away from too much corporate buzzword lingo that makes the vision unrelatable.

Leading Questions and Considerations

- What is our purpose?
- Where do we want to be?
- What does our future architecture look like?
- What outcomes do we want to achieve?
- What does success look like for our portfolio of products one year from now?
- What is painful today?

Suggested Tools and Tactics

- Portfolio/product community map (a Venn diagram map displaying product dependencies, customers, and stakeholders)
- Personas or customer descriptions (who are they; what do we know about them, their pains and joys; how can we deliver value to them; what assumptions are we making?)
- Outlook or outcome-based roadmap
- Product charter (what, why, who)
- Objectives and key results (OKRs)

Tactic 2 and 3: Map the Taxonomy of Products and Map the Technologies to Products

Value stream mapping, or experience mapping, is an ideal place to begin this conversation. Visualizing the customer experience end to end (from concept to cash, or product creation to customer delivery) will provide key insights into what products should be in a portfolio. Engage a coach or facilitator if possible to keep the conversation moving and neutral. A neutral facilitator can help drive collaboration forward effectively, especially when ownership battles or redundant capabilities begin to surface, which can happen quite often. Other supplemental approaches include leveraging existing service catalogs, configuration management databases, or business capability frameworks as relevant ways to kick off the product taxonomy definition. The key is to find a comfortable starting point to begin the product taxonomy definition conversation.

Who Should Be Involved?

- Technology and business leaders for the portfolio

Leading Questions and Considerations

- Why does your team/space exist?
- What capability or customer experience are you enabling or delivering?
- Who are your customers?
- Why do customers turn to your team for help? What problems do you solve?
- What are the applications or technologies that deliver or facilitate that customer experience?
- What are all the applications that your team is responsible for?
- Why do the applications exist? Do they each deliver a common experience or is there room for consolidation, sunset, or realignment of an application to a different team? What customer problems do they solve? How long have they been in existence?

Suggested Tools and Tactics

- Value stream map (on a whiteboard, collaboratively, face to face if possible)
- Value stream integration (implementation of the mapping exercise)
- Product community map (Venn diagram map displaying product dependencies, customers and stakeholders)
- Customer definition (personas)
- Product taxonomy document (see Table 2)

There are a couple of suggested approaches to defining your product taxonomy. Once the value stream map is constructed, additional layers can be added below the value stream to identify product groups, products, and applications. You can begin from the top at the portfolio level and work your way down to the applications. Start at the highest level, breaking it down into smaller chunks based on the value stream or customer experience. Leverage the leading questions above to facilitate the conversation. A recommended approach is to start at the application layer and work your way up to products (customer experiences) and product groups as in Figure 9.

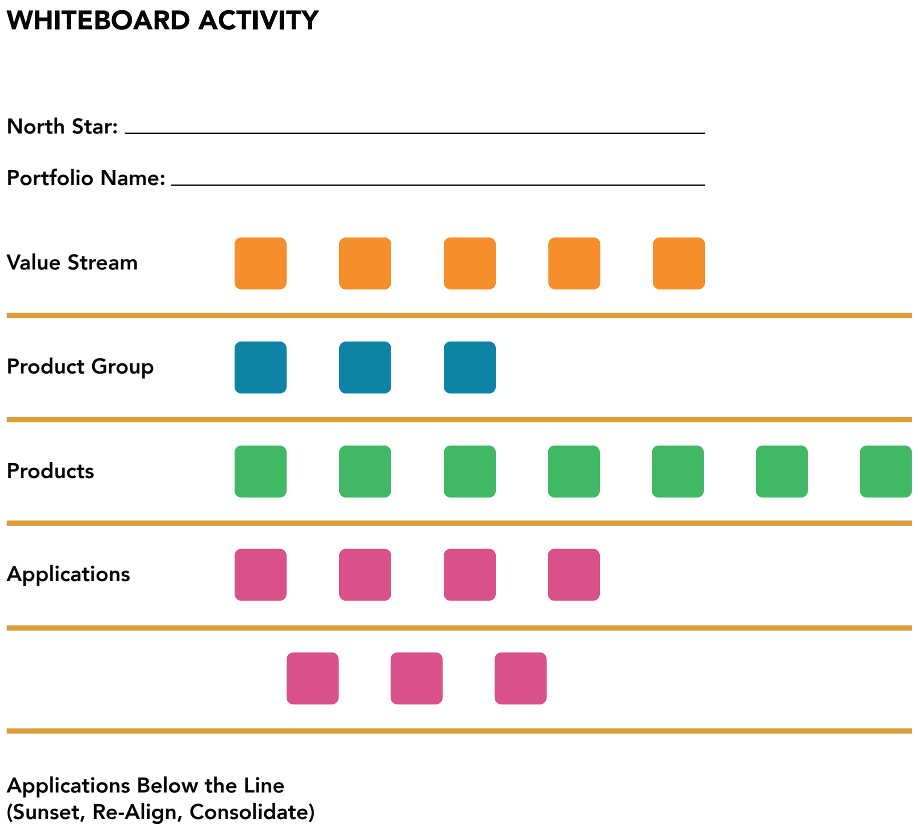


Figure 9: Whiteboard Activity

Tips For Creating a Product Taxonomy

- Use business-friendly language when naming your products. Do not use acronyms or ambiguous names. Anyone should be able to read your product taxonomy and understand the experience the product delivers.
- Ensure that you have defined both internal and external products and that each product has a customer specified.
- Focus on the business and customer outcome, capability, or experience the product delivers, not the application used to deliver it.

- Remember that products will likely remain consistent year after year while the technologies and applications will always change as new solutions are implemented.
- Ask the “5 Whys” to ensure the root of the experience is found; this can be a helpful technique when the application becomes the primary focus.
- Get feedback from your business partners, coaches, team, and peers.
- Expect an imperfect path. Pick a starting point and implement. Learn from the first iteration of the product taxonomy and be comfortable adjusting it quarterly or annually as needed.

Product Taxonomy Artifact

The product taxonomy should be a shared artifact that provides transparency across the various product portfolios. The artifact should be maintained by a centralized team to ensure consistency and safety of the information. Publish updates to the taxonomy on a quarterly or biannual cadence. Table 2 shows a template of a product taxonomy.

Portfolio (L1)	Portfolio Leader	Product Group (L2)	Product (L3)	Product Life Cycle	IT Leader	Product Owner	Business Alignment	Applications
Infrastructure & Operations		Network	Connectivity	Invest				
Enterprise Data		Data Science	Data Analytics	Sustain				
Security		Information Security	Cyber Security	Incubate				
Marketing		Loyalty	Rewards	Sunset				
Digital		Mobile	Cart	Invest				
Finance		Payments	Mobile Wallet	Invest				
Corporate Systems		Human Resources	Learning & Development	Sustain				
Etc…								

Table 2: Example Product Taxonomy Template

Tactic 4: Map the People to the Products

A well-defined product taxonomy will serve as a blueprint for how to redesign the organization around the products. A fundamental shift when transforming to a product-centric operating model is structuring teams to own the products end to end. In a traditional project model, the teams are organized temporarily to achieve specific project goals and then disbanded, leaving no formal owners for what was created during the project. At the heart of the product model is ownership, accountability, and empowerment.

Teams should be small sized (approximately five to ten people) and cross-functional, containing all the skills needed to plan, design, build, and run the product. Consider the number of product teams who will be needed to support a product. Ideally, the product-to-team ratio should be one-to-one; however, it is possible to have multiple teams supporting a product. To determine the number of teams needed, the scope and scale of the product should be assessed based on historical data, overall demand, complexities, and dependencies.

Assessing multiple teams against a product is part art and part science. It's recommended to start on the conservative side and scale to additional teams as needed over time. Additional details on product roles, product team characteristics, and other workforce considerations are provided in the Workforce and Talent section of this paper.

Who Should Be Involved?

- Technology and business leaders for the portfolio
- Human Resources

Leading Questions and Considerations

- What skills and roles do you have currently? Are there gaps?
- How many teams will be needed to support a product?
- Is the product manager/owner identified? Do they come from the business or technology organization? Why?
- Do you have traditional project family or functional roles (e.g., project managers, business analysts, process or functional analysts, testers) that need to be eliminated or repurposed into product roles (product owner, scrum master, UX, engineers)?

- Is there an opportunity to automate manual tasks (e.g., testing, processes, repeatable functions)?
- How many layers of management does your organizational structure have? Are there redundancies? Do you need to reduce the management structure to enable flow and team empowerment?
- Is additional training or coaching needed to skill-up in product management and modern engineering practices? How will learning opportunities be provided?

Suggested Tools and Tactics

- Full-stack product teams
- Skills matrix or assessment (map the skills and people you have and need and assess for gaps)
- Industry definitions of product team roles (anchor to these as much as possible, and don't try to create new roles that will overinflate the size of the product team as this practice will also make it easier to hire externally as needed)

Tactics 5 and 6: Identify any Gaps or Blockers and Establish an Action Plan

Transparency of gaps or blockers that could derail the product transformation is critical to transformation success. Decisions will need to be made regarding how to address the gaps and blockers. As each portfolio works through the process of mapping their product taxonomy and designing the teams, common themes may begin to emerge across the enterprise. These common themes should be addressed holistically to ensure consistency with the implementation of the product model.

Who should Be Involved?

- Technology and business leaders for the portfolio
- Teams
- Other key partners such as Human Resources, Finance, Learning & Development, Agile & product coaches

Leading Questions and Considerations

- What blockers, risks, or constraints do you have to work through to move your organization to a product operating model?
- Who can help with the solutions?
- Is the funding model product-driven or project-driven?
- Are there redundancies within or across the organization?
- Are there political debates about who should own what?

Suggested Tools and Tactics

- Establish an action plan to resolve the blockers and align clear ownership to each item.
- Manage the work through a transformation backlog or kanban board.
- Maintain transparency by keeping the blockers and actions visible to all involved
- Sustain momentum by forming a transformation team of leaders who will continuously work through transformation backlog.

Tactic 7: Coach and Accelerate New Product Teams

Once the teams are formed, provide them with coach support to launch and accelerate optimization. Additional guidance on how to coach and accelerate teams is provided in the Workforce and Talent section of this paper.

Who Should Be Involved?

- Technology and business leaders for the portfolio
- Teams
- Other key partners such as Human Resources, Finance, Learning & Development, Agile & product coaches

Leading Questions and Considerations

- What blockers, risks, or constraints do you have to work through to move your organization to a product-centric operating model?
- Who can help with the solutions?
- Is the funding model product-driven or project-driven?

- Are there redundancies within or across the organization?
- Are there political debates about who should own what?

Suggested Tools and Tactics

- A team of Agile technical coaches to continuously teach, mentor, and coach the leaders and teams.
- A product charter for the team to to clearly define the product, articulating a clear product description (or elevator pitch) of what the product is, the business/customer value it provides, and a clear definition of the customer the product serves.
- A product community map (stakeholders, customers, and dependencies) and personas.
- A team working agreement that defines the team's own routines, best practices, and agreements that will optimize their productivity.

Constraints That Need to Be Overcome

The product taxonomy is the starting point for shifting mindsets from a project and functional silo organization to a customer-focused product organization with clear lines of accountability. The exercise of mapping products drives leaders to think differently about their organizational structure, accountability boundaries, customer-focused outcomes, and overall alignment to business strategies. Through our interviews, we learned that the following constraints surfaced when the product taxonomy was missed or underutilized:

- **A lack of clear product definition and product mapping:** This leads to excessive overlaps or redundancies across teams. Establishing a clear product taxonomy within each portfolio enables transparency across the organization. Redundancies or overlapping accountability boundaries become more visible when all products are outlined in a centralized taxonomy. Having this transparency helps leaders see where they need to connect to resolve the redundancies.
- **Fixation on Agile frameworks without a focus on products:** This will delay or negate the shift away from the project operating model. Ultimately, project mindsets that focus on temporary teams executing on time, scope, and

cost will persist, and the full benefits of the product model will not be realized. An alternative is to empower teams to self-organize around the processes or frameworks that work best for them, rather than mandating a specific framework. Too much bureaucracy will derail the shift to empowered product teams.

- **Lack of clear product ownership and accountability:** This will eventually paralyze an enterprise due to internal politics delaying value to the customer. Without a clearly defined product taxonomy, redesigning teams to become full-stack product teams will be challenging, as there will be no clear boundaries around what the teams should own. Leverage value stream mapping and product taxonomy to drive conversations across conflicting product boundaries. Engage business and technology leaders to identify product owners for each product and empower them to fully define the product, scope, and boundaries.

Workforce and Talent

Product Transformation Guidance—Workforce Model

Figure 10 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Workforce and Talent	• Start with Agile and Dojo boot camps. • Ensure a strong Product Manager focus. • Define "Technical" and "Experience" based Product Managers. • Ensure a capacity-based model for work commitment.	• Focus on internal employee insourcing. • Reduce management layers and optimize team sizes. • Scale boot camps/ Dojos. • Reduce non-technical roles.	• Drive for very lean organizational structure. • Evolve Scrum Masters into coaching roles. • Break down/pivot PMO organization.	• Functional outsourcing. • Functional siloing across the technology organization. • PMO-driven transformation. • Providing inadequate training.

Figure 10: Workforce and Talent Indicators

Why Is This Important?

Traditionally, organizations have been based on hierarchies with decisions made at the top, with detailed direction flowing downward through multiple management layers to the teams. Traditional enterprise structures organized employees by functional specialty departments, each with their own work intake mechanisms, procedures, and goals. Work was heavily governed by a traditional Project Management Office (PMO) focused on prioritizing the work for temporarily constructed project teams. People would be partially allocated to a variety of projects as determined by their respective managers as they manage overall team capacities. The traditional workforce would often be heavily outsourced to contracted third parties to reduce the overall IT spend within an enterprise.

As we look to a more modern product operating model, many of these organizational structures require significant changes to fully realize the benefits of being a customer-obsessed, technology-driven enterprise. Frameworks don't transforms organizations. Building high performing teams and then enabling them transform organizations. Simplifying the management structures, minimizing control-oriented functions, and investing in growing your talent are critical to achieving this outcome.

When addressing these problems, far too many organizations try to simply adopt an Agile scaling framework without addressing these foundational elements, then they wonder why they aren't seeing the same improvement results that truly Agile organizations are able to achieve. Success comes when you have a committed and engaged workforce who believe in the transformation and in growing the skills and practices necessary to be high performing. This can also create a flywheel effect as top talent tends to attract top talent, enabling you to hire more skilled candidates than you may have been able to previously.

What Is It?

As we assessed the fourteen companies interviewed for this paper, a few common strategies related to workforce and talent emerged:

- Establish modern product roles and team structures.
- Upskill teams through immersive learning environments (Dojos), growth mindsets, and by establishing a learning culture.
- Focus on employee insourcing (versus outsourcing to third-party contractors).
- Drive toward a lean organizational structure by reducing management layers and non-technical roles.
- Break down or pivot the role of the traditional Project Management Office (PMO).

Traditionally IT teams have been organized by functional specialties such as front-end teams focused on user interface/user experience, back-end teams developing code, database administrators, testers, support functions, and a variety of non-technical project roles (e.g., project managers, business analysts, process and functional analysts, etc.). A full-stack product team brings together all the necessary skills into a team that will take full ownership of building and running the product. A suggested target team size is approximately five to ten employees that fulfill all key product roles as outlined below.

Product teams should be empowered to own the product end to end, meaning they will take full ownership of product planning, design, build, delivery, and support throughout the entire life of the product. The product team members must be 100% dedicated to the product and not partially allocated to multiple projects. Partial allocation creates context switching that is detrimental to a team's predictability and productivity.

To maintain an innovative critical edge, it is recommended that the product team members be employed engineers (insourced) and not temporary contracted resources (outsourced). This is critical to building a strong product engineering culture within the enterprise. Internal intellectual property will accelerate innovation and speed to market without dependencies on external third parties. Additionally, the most innovative ideas that will drive your business forward come from motivated individuals who feel ownership and a sense of personal connection to the solutions they create and the problems they solve for their customers. This connection is difficult to attain when the contracted engineer doesn't have permanent skin the in game.

Team Structures and Roles

During the incubation stage, explore the product team roles. Find a team or subset of teams to experiment with placing people with relative skill sets into the product roles. Demonstrating success with a smaller, cross-functional product team can generate the momentum needed to scale to other teams.

At a high level, the structure of teams following a product model consists of the following roles:

- **Product Owner/Manager:** The person responsible for discovering what is valuable, usable, and feasible for a product. The product owner owns the product vision, understands the customer needs, and drives prioritization and decision-making around the product requirements. The product owner ensures the team is building the right things. This role is often from a business organization, but may sometimes exist within the technology organization and are often referred to as technical product owners. The technical product owner is a more technical role focused on platform or IT-for-IT products. Product owners who originate or reside in the business are more focused on how functionality should be delivered for a customer-facing product.

- **Designer:** The role responsible for the usability of the product. With a more front-end, customer-facing product, this would typically be a customer experience (CX) designer, but it may be a more systems/technology architect for a back-end product or platform.
- **Engineers:** These are the team members responsible for the feasibility of the product; they ensure the right products are built the right way. The engineers are intended to be organized full stack and full life cycle. This means they have the technology skills needed across the front-end and back-end components of what they are building. They also have the skills needed to drive the planning and delivery across the entire value stream, including analysis, design, engineering, QA, automation, and other skills needed for the product.

Empowerment and decentralized decision-making are both foundational to a product team. Traditional hierarchies diminish the team's ability to fully own and run their product. Product owners have the accountability to ensure the team is building the right solutions for their customers and the team has the accountability to ensure they are building the solutions the right way, ensuring feasibility and user experience are optimized. Healthy product teams:

- Are passionate about the problems they are solving
- Spend time on iterative research and prototyping
- Focus on roadmaps around customer challenges, not features
- Commit everyone to solve customer problems
- See a release as a beginning, not the end
- Celebrate learning, even from failures
- Understand the competitor space
- See technology as a tool only and are focused on the outcome a technology can help create, not the technology itself

- Constantly renew their worldview and have a growth mindset
- See the product as part of a bigger customer journey

Other potential roles that could be considered as valuable in a product operating model include:

- **Value Stream Architect:** The goal of this role is to accelerate the delivery of value across the value stream and work with the product owner on product priorities. The value stream architect assesses value across the value stream, invests in reducing technical debt and building new features, and assesses risk, dependencies, and redundancies across teams within the value stream. Ignoring technical debt will have an impact on speed, quality, and happiness measures. This role is typically aligned to the product portfolio, working holistically across the value stream of products within a portfolio.
- **Scrum Master or Agile Team Lead:** Acts as a servant leader to the team, protecting them from distractions and blockers. The scrum master wears multiple hats, including teacher, coach, mentor, facilitator, and motivator. The primary focus of the scrum master is supporting the team's continuous improvement needs. One myth that should be dispelled about this role is that it is a meeting facilitator role. If the scrum master is focused only on facilitation, then the most important aspects of the role are being missed.
- **Enterprise Product Coach:** Provides knowledge, experience, and guidance to lead teams throughout the transformation. Moving to a product-centric model is likely new for everyone involved, so having a team of experienced coaches who can help people through the transformation is critical to success. Teams are naturally delivery-focused due to learned behaviors from a traditional project model. Project-minded teams are accustomed to delivering all planned requirements without scrutiny or exploring customer needs. These teams take orders and execute them without full empowerment to design solutions that directly solve customer problems. In a product-centric model, teams need to learn how to manage products through continuous product discovery and delivery. Product discovery is often an area where coaches upskill product teams. Shifting their focus to customers versus project deadlines, milestones, and deliverables

requires coaching guidance to learn how to fully incorporate discovery and delivery practices into routines. Enterprises will often enlist third-party consultant firms to define and implement the transformation, but it's the internal team of coaches who drive the day-to-day change within each portfolio. The Dojo (which is further explained in the DevOps Enterprise Forum paper *Getting Started with Dojos*) is a recommended coaching environment that enables safe experimentation, learning, and acceleration through change.

The role of the traditional PMO should also pivot and/or significantly diminish in a product-centric operating model. In the companies we assessed, this step typically occurred during the optimize stage of the transformation. Traditionally, the role of the PMO was to direct, control, and standardize project delivery across the enterprise. Heavy governance and oversight are prominent in a traditional PMO, which become anti-patterns in a modern product model. In some examples we examined, the PMO was diminished over time and then eliminated.

An alternative approach could be to transform the PMO to become more focused on enabling continuous improvement for the product-centric operating model. The PMO could become an ambassador for change by evolving its responsibilities to support an Agile and product-centric culture. The PMO team members could each be assigned to a product portfolio as a supportive partner to the product portfolio executive and product manager/owner roles. Reporting product metrics, financials, and key product insights at the product portfolio level can become a gap if there isn't someone in place to aggregate this data at the portfolio level. During the scaling stage of the product transformation, the PMO could play a partnership role with Finance to ensure funding for the product teams is enacted, moving the enterprise away from funding capital expenditure projects. The PMO can also assist with reporting value, ROI, cost of products, and cross-product portfolio-level cost and value insights. Additionally, they could modernize compliance requirements by providing product teams awareness to compliance needs and guidance on how to best comply (e.g., risk and compliance, audit requests, evidence, etc.).

However, this shift in PMO responsibilities can't be taken lightly. In many cases, while the PMO does attempt to shift to these new product-oriented responsibilities, its underlying mindsets and skills haven't shifted. The end result is that the same bureaucratic control-oriented practices quickly creep back into your product-centric model.

How Are Enterprises Achieving Success?

The partnership between technology leaders, business leaders, and Human Resources is critical when designing the product-centric workforce model. A workforce strategy that considers delayering management, insourcing talent, and converting traditional roles to modern product roles should be a collaborative approach between leaders and Human Resources.

Delayering or Flattening the Organization

Delayering or leaning out the organization is likely to become necessary for a product culture to stick. Delayering strategies tend to begin during the scaling stage and continue through optimization. Multiple layers of management approvals will paralyze a product team's innovation and speed to market. Product decisions become decentralized from the management layers with product accountability residing with the product manager/owner. Delayering is necessary initially to flatten the organization, removing multiple middle manager roles. Drive toward a leaner organizational structure with less bureaucracy by reducing management layers. In some of the companies we assessed, delayering reduced management from ten layers to three or four, significantly leaning out middle management.

The manager role changes significantly with an Agile and product transformation, shifting the role from directing the work to servant leadership as product strategy and decision-making become decentralized. Some middle managers will lean into the change, finding a new path for themselves as engineering managers or product managers. Others may opt out as they'll find the loss of control over their space and team to be frustrating. Even more common, middle management is most commonly cited across enterprise transformations as the stakeholder group that resists change the most, making it very difficult to drive these changes across the organization. "The frozen middle" is a term that best describes this problem and will be discussed further in the Culture and Leadership section.

Conversion or Elimination of Traditional Functional Roles

Traditional project family roles, such as project managers, business analysts, process analysts, functional analysts, and testers will need to be repurposed into modern product and engineering roles. Much like the delayering approach, role conversion typically begins during the scaling stage as product teams are being formed.

A strategic approach will need to be determined to gracefully transition existing people into new roles and/or hire externally to fulfill the necessary skill sets. In some of the cases we studied, this role conversion was approached through skill assessments and data gathering followed by a "big bang" approach of shifting people into new roles. This approach can be challenging if people aren't supported appropriately with proper training and onboarding into their new roles. An organizational structure that supports a product-centric operating model is one that eliminates functional silos, heavy PMO governance, and traditional paperwork-heavy hand-off processes.

Insourcing Talent

Outsourcing is often viewed as a strategy to reduce cost while also shifting responsibility to a third party. At the heart of a product model is empowering autonomous teams to truly own their products. You need your teams to have an owner, not renter, mindset. This is very difficult with outsourced teams.

Focus on growing engineering talent within the enterprise, nurturing an engineering culture. As companies scale the product model, insourcing becomes a great focus as skills are assessed and product teams are formed. The quality of code will inherently improve if engineers have long-lived experience with the product they own. In-house engineers will have a better understanding of the overall architecture and ecosystems within which their product lives. Teams made up of employees will have experience and expertise in managing the technical debt of a product, they will also have the motivation to build loosely coupled, extensible code and have the ability to proactively fix issues more permanently then a less experienced, temporarily (or shorter-term) contracted developer. The bottom line is that mindset drives outcomes.

Outsourced developers often focus on delivering with a project mindset, focusing on deadlines, cost, scope, and often even specific activities required by the terms of their contract. This tends to stifle innovation. An insourced team of engineers who own a product end-to-end will generally have passion, motivation, and creativity to innovate quality solutions for their customers. Obviously, it isn't realistic to insource all technical talent in most enterprises. While functional outsourcing should be avoided, staffing some roles on a product team with contractors to scale up the team can be a reasonable strategy as long as the team is still primarily employees committed to the product long-term.

In some of the companies we assessed, the engineering ratio of contractors to employees drastically changed during the scale and optimize stages of the product model implementation. In one enterprise, the IT footprint was over 10,000 people in IT; 70% of them were contractors and 30% were employees. Within one year, this footprint was reduced to approximately 6,000 people. This drastic change was achieved by a concerted effort to remove contractors, which pivoted the ratio to 70% employees and 30% contractors. A few years later they leaned out the organization even further, reducing the overall IT footprint to 3,500 people.

Upskill Teams

Once the product teams are formed during the scaling stage, they'll need support to accelerate and optimize their practices. There are a few approaches to accelerating new teams. One approach that has proven to be successful across a variety of enterprises is to establish a Dojo staffed with Agile, product, and technical coaches. A Dojo is an immersive learning environment where teams are guided toward new ways of working, leveraging Agile, Lean, DevOps, and product-oriented and modern engineering practices. Teams learn by applying new skills on real product work that delivers value frequently throughout their Dojo experience. The Dojo is a safe environment for teams to experiment, fail, learn, and adjust rapidly.

A few activities that are critical to team success as they initially form include:

- **Product Definition:** establishing a shared understanding and team alignment to what the product is; why it exists; the business value of the product; the life cycle vision for the product; who the product serves; the product community map depicting stakeholders, customers, and key dependency partners; the product's problem space; and the product objectives and success measures.

- **Team Working Agreement:** establishing team norms and agreements about how the team will work together (e.g., tools, communication practices, routines, the definition of ready and done, values, backlog practices, etc.).
- **Product Discovery:** establish an approach to quickly test product ideas against business, technology, and user experience to gain a holistic view of an idea, de-risking the idea and gaining immediate customer feedback through hypothesis testing and prototyping. Assume ideas are wrong until proven otherwise. Discovery helps the team determine the right products to build instead of building unnecessary and superfluous products. In a traditional project culture, we assume that our big ideas are correct and move them directly into a project for execution and delivery. Building without discovery is an expensive and slow way to learn that our ideas are wrong or don't meet the customers' needs.
- **Foundational Workshops:** establish Agile, product-oriented, and DevOps foundational training or workshops to ensure foundational mindsets are established consistently across the team. The team anchors to a mindset before practices or frameworks. This enables teams to safely experiment with frameworks to establish team optimization.

Constraints That Need to Be Overcome

Some of the key constraints that have been observed related to workforce and talent include:

- **Functional outsourcing:** This leads to a product team structure with an imbalance of employed engineers versus contracted third-party engineers, which puts innovation and competitive edge at risk. Focus on the outcomes you want to achieve as an enterprise and fuel those outcomes with insourced engineering talent. Invest in the company's innovation and future by growing an in-house engineering culture.
- **Functional siloing:** Another common failure for transformations is when teams operate disparately and have individual goals. In this scenario, the teams aren't dedicated and persistent. Employees often get conflicting directions from their functional managers and their product owners. To combat this, move to full-stack who that are truly organized around products and not leaders.

- **PMO-driven transformation:** Earlier we stated that frameworks don't transform organizations; building high performing teams and then enabling them transforms organizations. Typically, when the PMO is in the driver's seat and is asked by their C-level leader to transform the organization to Agile, they are essentially being asked to disrupt themselves, which generally isn't a recipe for success. The outcome tends to be a scaling framework layered on top of a project-centric organization. This minimizes the amount of real change in the organization and is incremental at best. It can even negatively affect team performance, as the team is now being asked to deliver in an Agile model while still being in a broader operating model and organizational structure oriented around waterfall project models.
- **Inadequate training:** When people have been repurposed into new roles and haven't been trained accordingly, it sets them up for failure. Support the teams by nurturing a learning culture. This can be achieved by providing continuous learning opportunities through formalized training and immersive learning experiences through environments such as the Dojo. Engage or establish an internal coaching team who can help accelerate upskilling product teams in product-oriented, Lean, Agile, and DevOps models.

There are many aspects to the workforce that should be considered when shifting to a product-centric operating model. The critical focus comes down to building small, empowered, autonomous, cross-functional product teams who have full ownership and accountability for their product end to end. Traditional functional specialty roles do not translate to a cross-functional product team. Build the teams with the right skills and roles, then trust and empower them to deliver value continuously.

Funding Model

Product Transformation Guidance—Funding Model

Figure 11 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Funding Model	• Establish pilot teams as capacity-driven. • Work within budget and financial constraints. • Use a quarterly funding model for piloting teams.	• Shift from project funding all teams for technology investments. • Release funding quarterly based on value delivered. • Create a "Strategic Investment Committee." • Epic-based funding/"Shark Tank."	• Continue to focus on alignment on business priorities and product funding. • Expand funding model to non-technology investments.	• Technology organizations are treated as cost centers. • Business leaders expect to fund features.

Figure 11: Funding Model Indicators

Why Is This Important?

In traditional project-based technology organizations, funding for the technology group is derived by executive leadership based on a prioritization schedule of big-ticket items stacked up against historical overhead expenditures. This budgeting exercise is a painful process and is nothing more than a wild guess at how much funding will be necessary to meet the needs of their business counterparts. This common practice results in management struggling to manage a budget that was set up to fail before funds were even distributed.

Unfortunately, most organizations continue to use this imprecise method of funding. An indicator of success in product-centric delivery is an increased focus on funding capacity and outcomes instead of blanket funding requirements.

All domains of product transformation require a large amount of fortitude and resilience from the transformers. Organizational inertia is strong in every corner of the organization, but altering the funding model may be the most difficult part of a transformation. Preconceived notions of how time is kept, how assets and projects are capitalized, and how departments are held accountable may be some of the most difficult structures to overcome.

With that being said, if you can work within the boundaries of your organization's current funding model to incubate the transformation, changes can be introduced slowly and show a great deal of success with the transformation.

What Is It?

The funding model is crucial to scaling your transformation. Although you can typically start a transformation and incubate a few teams within a traditional funding model, an antiquated funding model will create a huge amount of friction to the success of your transformation because of the difficulty in showing business outcomes and value from the product teams. Product-centric organizations hold leaders accountable by encouraging results that deliver value to both the business and the customer, whereas project-based organizations use budgets and timelines to measure success. The phrase "on time, on scope, on budget" has no place in a product organization. Product organizations are held to the standard of delivering value to the customer rather than simply meeting a set of requirements.

Through our interviews, we noticed that an organization's funding model could either be an impediment or a catalyst to shifting from project to product. Many responses highlighted the role of finance and funding in a successful shift. While the responses all indicated the importance of the funding model in the transformation, there was a wide range of variance in how organizations encountered and managed this domain. From this, we were able to gather the following practices and recommendations in terms of how the funding cycle interacts with the transformation.

How Are Enterprises Achieving Success?

It became apparent very early in our research that the funding model in large enterprises was typically one of the last domains to show significant change. Using the three-stage model of incubate, scale, and optimize, the incubation stage showed the least amount of change in the funding model across all of the transformation interviews. A good rule of thumb for the funding model in the incubation stage is to "fight winnable battles." Transformation teams used several strategies that became the incubate indicators:

- **Work within the budget and financial constraints:** This indicator suggests that you shouldn't try to fight the budget but use the existing structure to experiment and learn the mechanics of the transformation before biting off more than the organization can chew. Understand what constraints the current budgeting model may impose on your transformation and find ways to leverage the PMO reviews to show the value your transformation has and how you can continue to operate within those constraints, even though they can be cumbersome and cause unnecessary friction.
- **Establish pilot teams as capacity-driven:** For organizations with a more forward-thinking financial department that was willing to experiment, small teams were established and "funded" for a short time. During that time period, team leadership reported outcomes to the finance department to show value-driven success. This indicator often increased the velocity of acceptance among other teams when moving into the scaling mode of the transformation.
- **Use quarterly/biannual funding for piloting teams.** To introduce more accountability for product team leadership and show that accountability to your finance teams, reject the idea of the yearly budget and use a "prove my worth" model by requesting that funds be released to teams quarterly or biannually instead of annually. This practice requires the product teams to present their delivered value prior to receiving any funds.

Once organizations find traction in their transformation and move out of the incubation stage and into the scaling stage, they encounter a different set of indicators. From our interviews, the scaling indicators build upon the incubation stage, often solidifying and standardizing earlier transformation efforts. The indicators for the scale stage are:

- **Shift from project to team-based funding for technology investments:** Building upon some of the small changes made to funding pilot teams, this indicator pushes the organization further along the product model. If the technology organization can embrace the transformation, a good way to scale up the number of teams running a capacity-based funding model is to use the technology budget to begin the larger transformation.
- **Create a "Strategic Investment Committee":** Several of the enterprises we interviewed used a committee or board to distribute funds across the organization based on business cases. One of our interviewees described a committee with this function as the "Shark Tank." In the Shark Tank model, business and technology leaders present their epics to the committee. The Shark Tank committee asks the tough questions and, in the end, assigns funds based on the most compelling business cases.

Since the funding model domain seems to trail other domains in its ability to permeate the business-side of the enterprise, the optimize stage is where the business and technology groups start funding capacity and value as a single organization.

- **Focus on alignment between business priorities and product funding:** This indicator is important in the optimize stage because value is derived from business priorities. To truly measure whether value is being derived directly from priority investments, teams must correlate their capacity and cost of that capacity to the value derived from what the team delivers.
- **Expand funding model to non-technology investments:** This indicator is very similar to the previous one. As the transformation hits the optimization stage, the product funding model must be applied to all aspects of the business.

Based on our interviews, it appears that the path to transformation was idiosyncratic based on the specific enterprise. We noticed several organizations were in the process of either changing or reviewing how funding was provided and released to software and IT initiatives, and how value delivery was tracked by their financial partners. For example, the technology leadership team at one organization realized that projects and time tracking were an impediment to innovation. To combat the friction to innovation, the transformation team spoke to an external finance team at another large tech company who made the case that, at the accounting level, tracking time for software delivery projects was fundamentally flawed. This information motivated the organization's finance team to shift to a model that was shown to be more effective by a software innovator. However, when it came time to implement the model, there was no clear replacement within the organization, and the change was limited to select customer-facing teams. This is an excellent example of how changing the funding model domain can be very difficult at an enterprise level, requiring a large amount of political capital and partnership with the finance organization.

Another common trend was a lack of consistency in what was funded. Organizations that embraced Agile were exploring the funding of features and epics. Those who were deploying the "Spotify Model" were exploring funding teams, otherwise referred to as capacity. Others were exploring funding strategic initiatives and in one case OKRs themselves.

Some of the best-case transformation scenarios were also uncovered during this exercise. Indicators show the practices that most should follow and look to implement, but the scenarios below are unique to organizations fertile for more aggressive tactics:

- **Involve Finance from the start:** When making a shift that affects the funding model, make Finance a key stakeholder in helping define the end-state. The new model must support the way Finance will track value and delivery. Involving Finance at the tail end can result in a product operating model that is not aligned to business operations overall, which can stall or derail the transformation.
- **Define end-to-end value and cost metrics:** Finance already has methods for tracking the cost of development and other functions. According to Donald Reinertsen, author of *The Principles of Product Development Flow*, the most important part of product-orientation is that value streams are tracked for life cycle profitability.[2] If value streams are fragmented, end-to-end metrics cannot be tracked or funded, which inhibits the key shift of moving from a cost-centric

operating model to a profit-centric one. In *Project to Product*, the Flow Framework proposes the tracking of flow metrics and business results in order to provide one such anchor for the finance team to switch the funding model to be based on end-to-end product value flow and cost.[3]

Constraints That Need to Be Overcome

The largest and most prevalent constraints we encountered and documented in the funding model domain relate to how transitioning the funding model must at once overcome yet utilize the project-based funding gates that already exist.

- **Technology organizations are treated as cost centers:** In traditional technology organizations, the CFO and finance group see the technology investment as a pure cost center. In the 2018 DevOps Enterprise Summit London presentation "Project to Product: Practical Realities at a Large-Scale Enterprise" Carmen DeArdo contends that IT is viewed as a cost center with little to no capability to generate profit.[4] With this myopic thinking, it is incredibly hard for technology and product leaders to make the value of their teams, their products, and the financial investments visible. This paradigm is a bit easier in a product-centric transformation because product owners tie technology investments directly to customer value. To continue combating this constraint, ensure that the value being delivered is visible.
- **Business leaders expect to fund features:** In project-based organizations, finance and business leaders expect a yearly budgeting exercise where they dream up some grand opportunity and go for a "big bag" of money to fund their supposed great idea that will take all year to deliver. From earlier indicators, we know that project-based funding is not the path to success. The product transformation must make the value stream priorities visible and show business and financial leaders that the capacity funded through the evolving model continues to deliver value. To combat the single annual budgeting exercise, ensure that leaders see how changing priorities equate to positive value for the customer.

Our subjects demonstrated how important it is to involve Finance in the shift from project to product. Although there is clear guidance on how to fund project-oriented initiatives, the lack of a clear model for financing products can be an impediment to true product-centric transformation. Drawing from successful models at other organizations or establishing best practices, like flow metrics in *Project to Product*,[5] can help provide the finance team with a good foundation for change.

Architecture

Product Transformation Guidance—Architecture

Figure 12 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Architecture	• Establish flexible, low-friction ability to build. • Introduce architectural patterns that help with speed of delivery. • Engage enterprise and business architecture organizations.	• Create "toolchain as a product" paradigm. • Focus on automation. • Adopt to open-source platforms and tools. • Create architectural communities.	• Federate architectural decisions for speed to market. • Move to cloud and cloud native.	• Changing operat-ing model without deliberate focus on decoupling systems. • Bureaucratic architecture processes impede team velocity.

Figure 12: Architecture Indicators

Why Is This Important?

Architecture exists within applications, solutions, and the enterprise, whether time, effort, and resources are allocated to it or not. Architectures evolve over time just like most processes and organizational priorities. To enable delivery teams to operate at their highest velocity and remove as many impediments as possible, a focus on architectural improvement and streamlining your product delivery capabilities is crucial. You must ensure that the transformation evolves the architecture into a catalyst for desirable outcomes instead of the architecture devolving into additional overhead for teams to manage.

When it comes to a product model transformation, one of the most critical roles of architecture is to enable the product teams to work in an autonomous and decoupled way, minimizing dependencies and constraints on issues that are outside of a team's domain.

What Is It?

When defining architecture within an enterprise, most imagine individual applications or specific patterns typically used in developing software. However, architecture in the context of an enterprise product transformation encompasses all aspects of the business. Architecture is about describing, documenting, and transforming the enterprise's most fundamental systems.

The key to changing architecture in a product-centric organization is to find ways to leverage existing processes and structures while identifying future organizational goals, and to use the transformation process to enable and drive those architectural initiatives. Moving from project to product is about adding value. Find ways to use each aspect of architecture to aid the delivery of that value.

How Are Enterprises Achieving Success?

Like all transformation domains covered in this paper, organizations have varying levels of progress based on their stage of transformation. In several of the interviews we conducted, the enterprises were using their flexible architectures as pillars to build value upon, whereas other organizations found architectural changes to be a byproduct of the transformational effort. In all cases, architecture was viewed as an accelerator to the value proposition of the product transformation.

In our 2017 DevOps Enterprise Forum paper *Value Stream Architecture*, we introduced the idea of architecture patterns and tenets focused strictly on the value stream.[6] The premise of the paper was that the same amount of care and feeding that goes into writing software should be taken to deliver software. We identified several tenets that value stream architecture should focus on:

- Software architecture
- Transformation teams
- Risk assessments
- Organizational structure
- Reporting
- Training

- Tooling

To complement the value stream architecture paradigm, we suggested a new role within the organization to focus on these tenets of software delivery: a value stream architect. The value stream architect is responsible for streamlining the delivery of value for each value stream. Very similar in concept and deliverables, the indicators for architecture demand streamlining value delivery and measuring that value through transparent, highly visible metrics. Product-centric architecture requires focus on value streams and the existing architecture surrounding those value streams.

The architecture domain is no different than any of the other domains when entering and executing the incubation stage: keep it simple and confined to experiment, and prepare for scale. During our interview process, we found these indicators had the highest amount of impact during the incubation stage:

- **Establish flexible, low-friction ability for teams to build:** The goal at the heart of the product transformation movement is to deliver value to the customer as fast as possible. A simple equation is used to show the negative impact of delivering value to the customer late, called the cost of delay. This equation, outlined by Don Reinersten in *The Principles of Product Development Flow*, suggests that the sooner you can get value into the customer's hands, the sooner your organization will see value from the customer.[7] This indicator plays into that same theme, that in order to deliver value to our customers quickly, we can't continue to deliver software in a project-based method. Reduce the friction for our teams to deliver value to our customers quickly.
- **Introduce architectural patterns that increase speed of delivery:** Just like the previous indicator, this indicator is about delivering value quickly but from the software architectural perspective. Introducing progressive architectural patterns for delivery, like event-driven programming and API-first development, will help increase the efficiency of the value stream and enable seamless future additions to the value stream.
    - ***Event-driven architecture:*** An event-driven architecture is a software pattern that leverages a highly decoupled pattern of publishing,

subscribing to, and reacting to an event or a "state change" within a system. This pattern allows for loosely coupled systems and promotes the creation of quick changes and new features with very minimal risk to existing features and value.

    - ***API-first architecture:*** In an API-first pattern, APIs are designed and structured as the very first action in the design process. This practice greatly increases the feature or product speed to market by allowing developers to work in parallel but also reduces risk by clearly defining the product capabilities up front.
    - ***Domain-driven design:*** This is a design pattern that fits well with a product-oriented and Agile model as it promotes iterative evolution of the design, enables you to set clear boundaries around the domain context of the product you're building, and helps drive a common language for what is being built that even the non-techies in the team should be able to understand.
- **Engage enterprise and business architecture organizations:** An enterprise's inertia can often be mapped and defined from its enterprise and business architectures. Many of the enterprise's processes, practices, and policies are analyzed and documented by these two groups. In fact, these groups are typically responsible for building and managing the capability model for the enterprise, which we learned was often a key foundational artifact in guiding the product taxonomy work. By engaging these two architectural disciplines early in the transformation process, the transformation leaders can begin to get buy-in from these groups to plant the seeds of change and to also help the transformation team understand the complexities that shifting to a product-centric enterprise could impose. Strong enterprise and business architecture groups drive organizational change and are, by nature, change agents; the sooner these disciplines are involved in the transformation, the better.

As the organizations we interviewed shifted from incubating to scaling their transformation, we noticed that the indicators also shifted from increasing velocity in building software to standardizing and streamlining the delivery of value, also known as the value stream. These scale stage indicators focus not only on initially delivering value to market, but repeatedly and predictably adding value by reducing the risk and time it takes to deliver that value to customers.

- **Initiate a "toolchain as a product" paradigm:** Building a team and competencies to focus on the toolchains related to value stream delivery is key to frictionless value. Our interviews suggested that without a focus on the repeatability of value delivery, product transformations had a difficult time showing value.
- **Focus on automation:** Automation is standard practice and typically associated with DevOps, but DevOps and product-centric delivery go hand-in-hand and complement each other. In order to build predictability and remove unnecessary work, teams must automate everything from software builds and tests to deployments and feedback loops. This automation does not come cheap or easy. There must be a dedication and rigor to building automation into every unit of work. To see the results of a successful product transformation, automation is a must.
- **Adopt open-source platforms and tools:** Using platforms and tools purchased from corporations often leads to vendor or platform lock-in. Open-source tools and platforms typically offer the flexibility of being platform agnostic and can be used in very different environments. Open source is often a leading indicator of a progressive, forward-thinking organization as well. These tools and platforms typically focus on doing one thing very well and have large communities working to improve them, which allows for quick adaptation for integration into new and trending services and capabilities.
- **Create architectural communities:** Architectural communities help educate and dissolve a centralized architecture structure, moving architectural decisions closer to delivery. These communities help disseminate information, such as patterns, trends, and organizational strategies, to teams closer to the value streams. Our interviews showed that the most progressive product transformations also had a network of decentralized architectural decision-making, with some even leveraging an inner-sourcing model to drive architectural decisions and capture those decisions as source code. These communities become the foundation for that decentralization.

Once the transformation moves into the optimization stage, architectural tenants should be a foundational element of each of the value streams. The indicators derived from our interviews show an increased focus on decentralization and the adherence to technology to remove even more friction.

- **Allow teams to make architectural decisions regarding speed to market:** This indicator builds on the earlier indicator about architectural communities. In order to remove bureaucracy and move decisions closer to the problem, responsibility for making architectural decisions must shift to the team level. Enabling teams to make the architectural decisions that are best suited to their value stream helps to empower them and drives a much higher velocity of value delivery.
- **Move to cloud and cloud-native computing:** Another common pattern found in high-performing product transformations was teams moving their platforms to cloud technologies. Underlying infrastructure should be consumed as a service, as needed, and not be a burden for product teams to manage. Some companies achieve this through private cloud implementations of their infrastructure. However, the public cloud providers tend to be at the cutting edge of technology trends and feature delivery. Using capabilities provided by the big cloud players (AWS, GCP, and Azure) gives teams a huge momentum boost in delivery by creating self-service opportunities for technology enablement and cutting out the middlemen typically found in traditional technology organizations. Coupling these cloud capabilities with Twelve-Factor software design patterns enables product teams to build applications that are truly cloud native.

Constraints That Need to Be Overcome

We found two prevalent architectural constraints in our research and interviews:

- **Changing operating model without deliberate focus on decoupling systems:** Most organizations that struggled with the architecture domain did so because they had a very rigid monolithic system that didn't lend itself to product-centric delivery. In highly coupled and intertwined systems, it's difficult to deliver value quickly without a tremendous amount of risk. This constraint suggests that to reduce risk and gain efficiencies in the delivery of value to customers the best way to approach the transformation is to first start decoupling the systems into manageable "products."

- **Bureaucratic architecture processes impede team velocity:** In some traditional architecture organizations, architecture processes such as architecture review boards (ARBs) create unnecessary gates in product-centric organizations. ARBs and other centralized processes take decision-making rights out of the hands of the leaders closest to the value stream. The teams working on their given product understand the problems and issues with their product better than a centralized organization. A slightly more aggressive approach to this constraint is to break down the need for conventional architecture practices by suggesting and experimenting with a decentralized or federated approach that delivers decisions to architectural groups as needed.

Culture And Leadership

Product Transformation Guidance—Culture and Leadership

Figure 13 summarizes the key indicators that contribute to transformation success discussed in this section.

	INCUBATE	SCALE	OPTIMIZE	CONSTRAINTS
Culture and Leadership	• Focus on establishing community. • Create a bottom-up cultural movement. • Establish the proper incentives to enable empowerment and cohesion.	• Build culture of continuous learning. • Leverage internal DevOps Days and Dojos. • Use transformational leaders to establish relationship with "bottom-up" transformational movement. • Focus on executive immersion and learning. • Emphasize "failing fast" and team empowerment.	• Become members and influencers in external community. • Leaders teach practices to other leaders.	• Executive behaviors contradict cultural aspirations. • Not addressing resistance to change from middle management. • Existing talent not being on board, leading to attrition. • Leaders resisting increased transparency, wanting to keep IT as a black box.

Figure 13: Culture and Leadership Indicators

Why Is This Important?

A product-centric operating model requires high levels of empowerment, collaboration, and experimentation. As Peter Drucker is widely attributed as saying, "Culture eats strategy for breakfast." It's critical that you focus on leadership behaviors and fostering a culture to ensure that the organization can thrive as it moves to this new model. Core DevOps principles such as collaboration and empathy are necessary when an organization is striving to accelerate its ability to deliver highly targeted value to customers continually.

What Is It?

The culture and leadership topics have been covered extensively in previous DevOps papers. However, in our interviews, we wanted to focus on a few key aspects of the culture and leadership domain that we felt were critical to address as part of an overall project to product transformation strategy. As a result, we primarily focused on two specific questions:

1. What was done to align leadership and orient leadership behaviors?
2. What actions did enterprises take to drive culture change and upskill their workforce given the changing operating model and expectations?

Effective leadership is critical to the success of any transformation. In this context, leadership includes technology executives, business executives, and middle management. In the interviews, we asked what was done to align these different stakeholder groups and what specific leadership behaviors were focused on.

Culture is a frequently discussed topic in the DevOps community. The levels of collaboration, sharing, and learning orientation needed to operate effectively when you are optimizing your organization for speed often requires a significant cultural change. There are a lot of great tactics in the DevOps Enterprise Forum paper *Tactics for Leading Change* that will help you on this journey. Additionally, we will identify some key tactics that enterprises have been leveraging to ignite and grow their project to product transformations.

How Are Enterprises Achieving Success?

The best examples we saw were when enterprises were able to simultaneously grow strong communities through a bottom-up transformation while also connecting with a clear, top-down drive to transform the organization. In the incubation stage, the transformation would typically start with a bottom-up push for change. Pockets of change agents would start to connect within the organization and would band together to start challenging preconceived notions for how teams were supposed to work. In some cases, these groups would even take a bit of a rebellious approach—

raging against the machine, so to speak—and attempt to distance themselves from the enterprise.

While that can be a good initial approach, bottom-up change can only go so far. A bottom-up movement is unlikely to reach all leaders and stakeholders across the organization. Successful transformations connect the bottom-up incubation with top-down, C-level buy-in. The key is to find a transformational leader with significant political clout and savvy who is willing to sponsor and drive this transformational movement. In some cases, enterprises were able to achieve this when new C-level leaders joined the company. Often when new executives join, they are looking to leave their mark on the company. If they are transformational leaders, they will typically look for a high-energy change initiative to sponsor and accelerate.

In one case, the chief technology officer who had joined one of the enterprises we interviewed had previously worked at a technology product company. She had a clear vision for the change she wanted to effect based on her previous experience. She personally led and sponsored the overall product transformation, including managing the expectations and commitments with her business counterparts, effectively pulling them in to be part of this change.

It was not uncommon to hear that before there was a driver for change, most of the companies we spoke with felt comfortable with the status quo. In some cases, this was coming from the top down. Some companies had extraordinary changes in market conditions which forced change onto them, but many of them were still content with their current state.

For some of the best change examples, this convergence of bottom-up and top-down transformation started to thaw the "frozen middle" that we referenced previously. In nearly all cases, we learned that one of the primary leadership challenges with transformation was middle management.

There are a lot of people in middle management roles in these large enterprises. These people have typically had success being operationally focused leaders. Additionally, they tend to face a lot of different demands in that they need to manage the expectations of the executives, their teams, and their clients. With everything they are responsible for, it shouldn't be a surprise that this group is often slow to warm up to new ways of working. People are more open to change when they see the net benefit to that change. They need to see a pathway for future personal success. People are more likely to reject change if they perceive too much effort, time, or risk is required for the amount of value that will be returned.

Some of the key indicators we saw enterprises follow when overcoming inertia were:

- **Define new incentives:** Early in the transformation, often while still in the incubation stage, executives would start to define a new set of incentives for their management teams. Not only would these incentives focus on outcomes over outputs, but they would also enable cohesion and empowerment. Examples of this would be encouraging management to create space for their teams to participate in learning- and community-focused events. Another would be encouraging reuse and sharing of code through inner-sourcing versus everyone building their own code from scratch. Identify the behaviors that you want to see and communicate these expectations to your teams.
- **Accept and acknowledge failure as part of the transformation process:** When scaling the transformation, leaders would publicly emphasize "failing fast" and team empowerment. These are often two very difficult behaviors to get leaders and teams to change. For many enterprises, people are not comfortable admitting failure. Executives need to highlight the learning opportunities that emerge from failure.
- **Teach new leadership behaviors through workshops:** To drive toward more servant leadership and empowerment-oriented leadership behaviors, some enterprises established leadership immersion workshops to expose the frozen middle and executives to this new way of working. They emphasized how teams are expected to work in this new model and provided real-world insights into why teams needed to be empowered to experiment and learn. Teams must continually discover and adapt the product they are delivering and increase their understanding of the viability, desirability, and feasibility of their project's value for their customers.

Across the enterprises we interviewed, there was a clear theme of focusing on growing a continuous learning culture. It's important to use community-based learning to educate workers on new practices. There are a few ways to do this:

- **Organize teaching workshops led personally by management:** In the early incubation stage, companies need to teach their workers about DevOps, Lean, Agile, and product-oriented methods. Success was seen in multiple instances where leaders took on the role of teachers and didn't simply mandate change. In one scenario, a middle manager personally facilitated and ran Lean workshops where he taught other middle managers how to behave and operate differently. By taking the time to teach them, he was able to achieve more alignment and commitment than if he had just asserted to people that they need to change. Don't just tell people what to do; give them examples and tools to teach them how to think.
- **Organize external DevOps/Product conferences:** These conferences are structured with the goal of encouraging people to connect, share, and learn by facilitating horizontal interactions across the company. Many companies modeled these conferences after the public DevOps Days, as these events have proven highly successful at building community. These internal events allow employees to present to each other in a fun, engaging, high-energy environment. Structures generally follow a more "unconference" format to make the events feel informal and inviting.
- **Host hackathons:** These have become a common tactic for enterprises looking to strengthen their engineering culture. They are yet another powerful way to bring people together from across the organization and get them to connect, share, and learn from each other. They are often focused on innovating on a customer or business problem that could be solved. There are many forms of hackathons in the industry, but we won't be covering that in this paper.
- **Utilize Dojos:** While Dojos have already been discussed due to their role in upskilling talent on modern product and technology practices, they have also proven to be a powerful mechanism to change culture in the companies that have adopted this model. So, how do Dojos enable this culture change?
  - ***Space:*** You can create a space to specifically foster openness, collaboration, and sharing. You break down the walls and create a fun and engaging environment that inspires creativity. Create teaming spaces that encourage different groups to come together, demo, and share their learning with each other. In more than one company, Dojo participants

commented that they feel like they are working at a different company due to the different experiences they have in the Dojo.

- ***Process:*** You can orient your processes in the Dojo to get teams comfortable making decisions for themselves. The Dojo can be a safe zone. People experiment, fail, and learn in an environment where such actions are explicitly allowed and encouraged.
- ***Coaches:*** Dojo coaches are often cultural experts and evangelists. They know exactly what is wrong with the existing culture within the company and are very well positioned to guide participants through the things they need to do to change. They also become a voice of the teams in the Dojos and coach management on what they need to change to enable and empower their teams.

There were not many enterprises we talked to that had truly optimized their culture and leadership. However, the few that were had become thought leaders and influencers in the external community. They would tell their stories and share their successes externally through conferences or outreach with other companies. Additionally, they would invite others to their companies to participate in their events and experience new ways of working.

Constraints That Need to Be Overcome

Culture and leadership are incredibly difficult aspects to change during a large-scale transformation because at their core these aspects deal with people and the interesting and challenging ways they react to change. While there can be a lot of constraints to work through in this domain, here are a few key ones:

- **Executive behaviors contradict cultural aspirations:** This can be a very difficult problem to solve. Sometimes executives will communicate the goals of the transformation, such as “we are looking to empower teams to make more decisions” or “we need to start embracing failure so we can improve learning.” However, the behaviors of these executives don’t change to support these goals. Examples of this include executives still making technology decisions without engaging their tech workforce, or overcommitting delivery for their teams while expecting teams to do “death march” releases, leaving no time for community engagement and learning. This is a huge barrier to change as employees don’t believe the executives and stop listening to what they say. Instead, they watch what they do. It can be especially frustrating for employees when an executive’s words are not in sync with their actions. The best way to address this is to get transformational leaders in place who will model the behaviors needed for the transformation. When a change in leadership is not possible, invest in executive-level coaching to help identify when these behaviors are happening and to guide executives on how to act differently. It’s also beneficial to define new leadership behaviors and expectations for the organization.

- **Middle management resists change (the “frozen middle”) and is not addressed:** This stakeholder group loses a lot of control in these new operating models. Technology tends to become much more transparent in a product-centric model. Not all leaders appreciate this transparency and would prefer that IT stay a black box to the business. Special attention should be placed on middle management to help them understand the value of these transformations, what their roles will entail, and how they can still thrive and be successful in the future model. You can leverage some of the tactics we already described in this section. When you’re incubating your transformation, focus on leaders who want to change and help them. As you build momentum, you can pivot more of the later adopters who want to see some success before they jump on board. However, even after all your best efforts, there might still be some that refuse to change. Build enough change momentum for leaders to realize that they need to decide to either get on board or leave the company. Some will leave, and that’s okay.
- **Existing talent does not participate in the new processes, leading to attrition:** A big aspect of this change from project to product is the move to full-stack teams who build and run products. This has a few implications on people. First, in traditional enterprises people focused on building depth of expertise in one or two skills. If things fell outside of their knowledge, it was some other person’s responsibility. In a product-centric model, a team is collectively responsible for the end-to-end product. This means people sometimes step out of their comfort zone to learn and contribute in an area of delivering the product that isn’t their primary expertise. Second, in this model teams

typically are responsible for running and supporting the products that they build and deliver. This means developers may start carrying pagers, which isn't something many have had to deal with in the past. Don't underestimate people's resistance to change. Some may even choose to leave as they don't want to sign up for these responsibilities. The biggest step you can take to help people through this change journey is to invest in them. Help them learn new practices, like how to engineer quality into the product they are building. Once they learn how much better the product outcomes can be following modern engineering practices and they feel proficient in doing these things, many will feel much more comfortable with the change.

Case Studies

Case Study 1: Retail Enterprise (Fortune 100)

This study focuses on a *Fortune* 100 retail enterprise that demonstrated sustained progression through the three stages of enterprise product transformation.

Transformation Implementation

Their journey began with grassroots experimentation in a small number of technology teams within the digital and API platform organizations. As initial success was seen in this experimentation, they also spanned their new practices across a few mainframe and monolithic applications. The initial focus of the incubation stage was on DevOps and Agile. A Dojo was established to demonstrate and accelerate the full-stack product model. Coaches supported immersive learning experiences within the Dojo and quickly educated workers on the new product team roles (e.g., product owner, scrum master, full-stack product team operating with an Agile mindset while practicing DevOps principles).

The success and learnings from the incubation stage quickly led to a "big bang" implementation during their scaling stage. This shift was directed by the CIO, who had previous product experience that crystallized his vision of a transformed organization. The CIO and various other leaders held frequent informal town halls to transparently communicate vision, intent, and approach as the model was being developed. The transformation implementation was led by internal technology leaders. This organization shift was implemented in waves over a nine- to twelve-month period. The number of projects this company worked on during any given time reduced from more than 800 to fewer than twenty.

The full organization was transformed during the late scaling stage and optimization stage. As the product model matured within the IT organization, the business realized that the formerly slow "black box" IT was no more. IT was now moving faster than the business, increasing their engagement with the public and igniting interest in adopting Agile and product-oriented methods across the business. The Dojo, which once served only IT workers, became the learning and acceleration environment for business teams to rapidly adopt Agile, Lean, and product-oriented concepts. The Dojo engagements were leveraged as opportunities to bring IT and business to the table together, further bridging the gap between business and IT. It was at this point that full enterprise or business agility became a real possibility.

Business and Technology Synchronicity

Product leadership roles were established within both IT and the business to encourage continuous alignment to products. As the product model scaled across IT and the business, the need to implement shorter planning cycles surfaced. Quarterly product planning (QPP) was implemented as a method to synchronize cadence, OKRs, and roadmaps of product teams. QPP was intentionally designed to counteract heavy bureaucratic governance. Product teams were educated on OKRs and provided guidance on how to form product networks in a self-organized way. Product owners were encouraged to connect with other product owners who were aligned to a common value stream, customer experience, or shared objective. Coaches provided facilitation support to help form product networks and establish product owner routines that focused less on the current sprint and more on removing escalated blockers, synchronizing OKRs, and managing dependencies. The principles of QPP were based on empowerment, collaboration, and Lean processes. As the transformation evolved into the optimize stage, the product networks became less dependent on coach support and more self-sufficient.

Operations was also redistributed to imbue support skillsets within each product team. This further underscored the full-stack product team concept of having full ownership of all design, build, and run of their products and scaling DevOps across the IT organization. This shift ultimately leaned out the support organization to include a technology operations center, which functioned as first-level support for call centers, routing incidents, and problems to the proper product teams.

Product Taxonomy

Technology leaders from each portfolio drafted the product taxonomy and mapped the entire organization to the product taxonomy structure. The taxonomy continued to evolve quarterly with official updates published biannually. Once the taxonomy was drafted, the organization aligned products and their supporting technology/application accountabilities to full-stack product teams. As product transformation matured into the optimization stage, the product mapping process also matured by leveraging the Dojo to host coach-facilitated product definition and discovery workshops. The workshops offered product teams the opportunity to more crisply define their products through product chartering and value stream mapping. It was important for the product teams to understand the larger ecosystem within which their product would sit. This helped prevent the product model from becoming bloated over time.

Workforce and Talent

Product owners were initially identified from within the IT organization then later transitioned into the business. Traditional project management roles (e.g., project managers, business analysts, process analysts, testers, and functional analysts) were replaced by product owners, scrum masters, and full-stack engineers. This shift also triggered a large-scale effort to delayer the organization, leaning out the hierarchical structure that had previously supported more traditional bureaucracy. The delayering reduced the management structure from ten layers of management to four. Prior to the product model transformation, the IT side of the business was primarily outsourced and offshore, with a mix of 70% contractors and 30% employees. Within the first year of the product transformation, this ratio flipped to 70% employees and 30% contractors. This was largely achieved by removing thousands of contractors from the workforce, driving the total technology workforce from over 10,000 people to approximately 6,000. Keeping the intellectual property in-house, created by employed engineers rather than outsourced to third parties, also preserved the enterprise's competitive edge. The organization was able to perform better on speed, quality, and cost due to improved prioritization and technology practices across the organization.

The experiential learning approach of the Dojos had proven to be very effective at "leveling up" people through the transformation. Thus, Dojos were scaled out to support the scaling stage and ultimately supported over twenty-five teams at any given time. This approach was coupled with Agile training and boot camps to help train teams as they were moving to the new product-based model.

Funding Model

Another critical enabler of the product model transformation was a "big bang" implementation of how work was funded. The funding model shifted from funding projects to funding persistent product teams. Funding evolved into a biannual process focused on value-based outcomes in the form of OKRs. Other aspects of the funding request would support any necessary software, hardware, and product team personnel. Leaders would need to determine how many product teams would be needed to support a product and build the team needs into the funding request. Funds would be released to teams quarterly (as needed) based on outcomes achieved, product roadmaps, and projected OKRs.

Architecture

Additional engineer empowerment was manifested by a quarterly CIO workshop, which gave the lead engineers a voice in architectural vision and decisions. Culturally, there was a principle shift to valuing guidance over governance regarding preferred technology strategies. Monolithic applications were no longer the norm. Priority was put on descaling monolithic, internally focused systems into smaller pieces that could be run by small, self-managing, customer-focused teams. Enterprise-class APIs and microservice strategies were leveraged to build common services and decoupled systems. Automation-first principles were of highest priority for all teams for all aspects of their product delivery end to end. Other critical functions were also automated into delivery processes, including Information Technology Infrastructure Library (ITIL) processes, audit evidence, and security controls.

Culture and Leadership

Significant focus was put on building an engineering culture that stressed the importance of internal intellectual property and inner-sourcing. To accomplish this cultural transformation, internal events such as hackathons, DevOps days, product and engineering inner-conferences, and demo days were implemented, which provided the employed engineers venues in which they could network, share knowledge, and learn best practices. Additionally, Dojos began to serve as a powerful cultural hub within the organization, enabling teams to start learning and practicing new behaviors in a safe environment. Previously, engineers did not have a voice or autonomy to creatively solve customer problems; rather, full solution requirements were dictated to them via waterfall projects.

Leaders were put through Agile mindset classes and immersive workshops on team functionality in the new model. This helped them adapt their leadership styles to enable and empower their engineers in this new model. A DevOps summit was run within the company with all technology executives and middle management invited. Speakers from the DevOps Enterprise Summit who were recognized for leading transformation within their own companies were invited to share their stories at this event.

While the product model implementation was viewed as a success in short order, it wasn't free of challenges or fail points. Some of the challenges included resistance to change from the frozen middle, or middle management. Agile and product-oriented methods were often viewed as a phase that would pass. Shifting people into new roles (e.g., project managers becoming scrum masters, business analysts becoming product owners, configuration analysts becoming engineers) presented failure points as people weren't provided adequate training to shift successfully. This gap led to attrition and frustration within teams.

These constraints were managed and addressed as the company continued to optimize their product transformation. Over time, they have been able to hire high-caliber talent who came from digital-native companies such as Netflix and have been able to continue driving toward a more Lean, highly efficient organization.

Reflection

It's safe to say that the typical success rate for a "big bang" transformation of this scale is likely in the low single-digit percentages. However, this company was extremely successful. Why? Upon deeper inspection, we uncovered the following characteristics that were likely significant contributors to their unlikely success.

- They had a burning platform, and everyone knew they had to change. Their industry was going through significant disruption. New threats emerged from Amazon, as they had a more digitally oriented business model and were quickly taking market share from other players in the industry.
- They already had a highly collaborative culture, making it easier to get their technology workforce to share and learn across the organization.

- They had fostered a strong bottom-up cultural movement as part of their DevOps transformation prior to kicking off their product transformation.
- Their entire C-suite turned over in about a twelve- to eighteen-month time-frame. New executives were looking to champion change initiatives. Within IT this enabled bottom-up initiatives to meet top-down initiatives, which accelerated the overall rate of transformation.
- Their new CIO had previous success with driving DevOps and Agile transformation in a previous enterprise.
- They had strong leaders driving their transformation who understood the internal environment and how to influence change across the organization.
- Their people were all colocated versus being geographically spread across many locations. This allowed them to drive more cohesion and connectedness as they formed product teams across the organization.
- Their business and security executives were willing to embrace a new model of working with IT to get work done.

Case Study 2: Apparel/Sportswear Enterprise (Fortune 100)

This study focuses on a *Fortune* 100 apparel/sportswear enterprise that had success incubating their model and is currently in the scaling stage of enterprise product transformation.

Transformation Implementation

This enterprise started incubating their transformation in their digital and customer-facing portfolios. They had strong leadership there that had previous experience moving to a product-centric model and was willing to champion a different way of working. This leader prioritized Lean value streams for their products with a relentless focus on continuous improvement. She had built strong relationships with her business partners who believed in similar outcomes. This allowed her to build momentum quickly. She had to be bold in the early stages. For example, there was a SAFe transformation underway across the organization, and she felt strongly that it wouldn't have the same impact as the Lean value stream or continuous improvement model. She decided to identify candidate teams across the portfolio who could try value stream mapping as an alternative to SAFe.

As this enterprise moves into the scaling stage, they are in the process of rolling this change out by portfolio, prioritizing the ones that will benefit most from the product-oriented model. They are lookeing at the maturity of each portfolio as well as which have the most dependencies on the transformation agenda. Based on that, they will prioritize their inventory/order management and enterprise data and analytics portfolio.

To guide them, they built a decision framework and playbook. Their scaling strategy was to essentially treat the transformation as small consecutive experiments: Assess team progress, evaluate whether what worked for one team would work for others, and validate whether it does. If it doesn't, modify and try again.

The communication strategy was at a corporate level, not just within the IT organization. Their overall business strategy had changed to be more direct to customer, and it was important to align their overall technology transformation strategy, including the move to the product-oriented model, accordingly. They are currently on schedule to implement this new operating model across their technology organization in summer 2019.

This initiative is championed by their COO and a cross-functional steering committee, including business leadership. Many forums have been held to promote the new strategies. The details of how they get to outcomes may change, but their overall goals do not. Having said that, one challenge they have is that some teams are behind on the current messaging. This can be confusing, as teams hear different messages across the broader organization.

Choosing what to measure at scale can be a challenge. This enterprise is currently trying to optimize for speed, but defining what that looks like can be challenging. They are focusing on the measurements defined in *Accelerate* to start. This includes lead time, deployment frequency, MTTR, and change failure rates.

Business and Technology Synchronicity

At the company level, these efforts were branded as "digital transformation." Given the shift in business strategy to be more direct-to-customer, the technology arm of the enterprise needed to be acknowledged as a strategic enabler of the transformation and not just a cost center. The enterprise is currently hiring their first chief digital information officer for all of technology. This company has recognized that this isn't just a technology transformation; it is a business transformation.

This is a highly complex transformation because there are many moving pieces. With any shift in company strategy, there will be mindsets that resist change. Sometimes this is due to a lack of clarity on what people are trying to optimize for. Their new approach requires a bias for speed, but there are a lot of people who still focus only on efficiency and cost. This creates a bit of a culture clash within the enterprise between the champions of change and those content with the old model.

The entire company is now on the same performance-based incentive plan. When the company wins; everyone wins. OKRs are leveraged to drive alignment with stakeholders. Every quarter they review the OKRs. If they aren't trending correctly, they shift. The intention is to have shared outcomes on both the business and technology side. They are currently adjusting their primary outcome from revenue to the actual resilience of the product. With this shift, if a product isn't performing, the business leaders (not just the technology leaders) will be accountable for the business outcomes and technology outcomes. Further, a product leader will be just as accountable as the technology leader for prioritizing tech debt reduction, security, privacy, and observability. The product team will still own all these responsibilities. Their head of product has also advocated for these changes as there is a belief that the product team should be just as passionate and accountable for technology requirements as they are about driving features. Organizationally, this enterprise is establishing dedicated business capability owners who have a technology counterpart. They strive to have a one-to-one relationship between business product and technology leadership.

Product Taxonomy

This enterprise recognized the need for a common taxonomy as the primary organizing construct for its operating model. Everyone talked about product differently. Within this company, the term "product" usually meant what the customer was purchasing. In building their taxonomy, they leveraged their service catalog as primary inputs for the services, applications, and technologies. At this time, the enterprise is still calibrating and reconciling their catalog with their taxonomy. For example, the company's search feature and their mobile app are both considered products.

However, there is discussion as to whether their system application (SAP) is a product or if it belongs in a different type of category altogether given that it is heavily packaged software. Defining product, platform, service, and capability models aren't always straightforward.

By using capability as a method of categorizing, the enterprise has been able to identify areas of redundancy that could be consolidated. For example, they accept orders in store, on a digital platform, and through partnered platforms, and previously had distinct products for each. They now have one product with three distinct experiences. They achieved this by breaking each product down into capabilities that provided actionable guidance for the distinct experiences.

Workforce and Talent

This company has established the product manager title in IT and business. The product managers in business focus on the customer experience whereas the technical product managers work on building the product. For external customer-facing products, having the experience-focused product managers in business makes sense. However, the more back-end, platform-oriented products are highly complex technically and often are not intended for customer interaction. In this case, the technical product managers own the business logic, flow, and the business rules that need to be enabled in order to realize desired outcomes.

The enterprise leaders are growing product management as a competency and career path. Also, recognizing an anti-pattern that often happens in these transformations, they are working hard to avoid taking internal candidates and moving them to another position or department without properly investing in them. Too often companies will give project managers or business analysts new titles and responsibilities but not give them the scaffolding to operate with a product mindset. The transition to product management comes with expectations. Internal candidates interested in transitioning to a project management position participate in a boot camp and extensive training process. Even with these structures, the transition can be tough. The enterprise is also growing their external talent in this space, but it is a highly competitive market.

This company is also making it a priority to increase the visibility of their work and track the reality of their work. All product team structures are visualized and observable. This visibility drives a conversation around priority and commitment. If a team is heavily indexed by operations work, for example, and the product manager is frustrated by the team's failure to meet goals, they can use their dashboards as a fact-based way to understand why. The product backlog reflects the different work types they track, such as features, tech debt investment, operational work, unplanned work, and discovery. To aid visibility efforts, the enterprise aims to achieve a ratio of six engineers to one product manager, but this ratio is somewhat variable, and in many cases a product is too big and needs to be broken down across multiple teams.

Funding Model

The current funding model here is similar to what most enterprises use. This enterprise has a baseline budget, a set level of funding to match overhead costs. Then there are strategic enterprise initiatives that are funded above and beyond baseline. Beyond that, the business groups also have some discretionary funding they can use to fund their individual priorities.

The enterprise leaders' goal in the scaling phase of their transformation is to move toward funding teams and products and away from funding projects. As an early move, they will stop giving business groups their own discretionary funding in summer 2019. Funding will then be centrally managed and aligned to their priorities. This will be a huge step toward transforming their operating model.

In the new model, initial funding to cover organizational capacity will be released at the beginning of the fiscal year in alignment with identified priorities, and then funding will be released quarterly based on a team's results and learnings. The enterprise created a strategic investment committee to govern the investment process, including the outcome validation.

Architecture

Within this company, the business architecture team and enterprise architecture team drove the creation of a full capability map. This served as a base for mapping and organizing their products and product teams.

In a lot of IT organizations, enterprise architecture decides centrally on new technology decisions, such as build versus buy. However, for their transformation, they felt this was not the right model and are putting the accountability and ownership of decisions in the domains. This aligns with some of the broader information we have received through our interviews as well.

Culture and Leadership

This company is transitioning to an outcome-based culture. Enterprise leaders agree that they need to move faster and use a better, more predictable delivery engine. With this goal in mind, leaders are also in agreement on needing shared outcomes, strategic prioritization, and a capacity-based team model oriented around products.

The goal at this company is to define outcomes across the organization and adopt a capacity approach with transparent work processes. Often, business stakeholders are not privy to the inner workings of a team's process, and making such processes transparent can build understanding and trust between stakeholders and engineers. This company is currently creating a model where key stakeholders agree on shared outcomes and strategic prioritization.

Reflection

This enterprise is beginning to scale their product transformation. The enterprise has a strong transformational leader who has been a catalyst for change. As her role expands, she is driving this model across more portfolios. A business strategy to drive a customer-centric model is being implemented, and there is clear alignment across the business and technology arms of the enterprise to invest in building product competencies. Additionally, there is a clear vision and plans for the funding model, which often doesn't occur until further into a company's transformation.

The enterprise is entering a very difficult scaling phase with a number of organizational change decisions looming in the coming months. This will be a very risky phase for the change program as leadership roles shift within the organization. The enterprise will soon have a new chief digital information officer, which will have a major impact on the direction of this transformation. As we learned through our interviews, new C-level leaders do one of two things: they either accelerate change or destroy it.

Case Study 3: Global Media Service Provider

The case concerns a multinational corporation providing business support systems via a SaaS model (SaaS-based multi-tenant back end). The most important product is customer care systems to large service companies in the healthcare and telecommunications markets. The company is by design product-centric (workforce management, customer care, billing, order management, messaging). It is heavily standardized, but products are customized for larger customers. This foundation made the scaling and sustenance of the project to product transition much more manageable.

Transformation Implementation

The project to product transformation was built upon a DevOps organization that had already existed for several years. This provided a solid base of expertise, and each DevOps team was already aligned to product teams in the business. As the project to product shift took shape, several modifications were made to the organization. The 500-strong engineering team, 200-strong operations team, and further 200 IT professionals were merged into a single organization. This brought together the build and run competences, and further strengthened the focus on the product.

A more difficult transition was moving away from a more traditional PMO to a portfolio leadership group. In fact, this only happened after the CIO was replaced. In the new setting, the project management capability is embedded in the product group. In addition to project management, the group also looks at investment and the communication plan. Moving away from the old mental model required a lot of "soap boxing" to sell the new journey.

As our interviewee stated, "We realized that this wasn't just a change of nomenclature. It's also behaviors and ownership. When you switch to a product focus, you are paying for a product backlog in perpetuity until you sunset the product. That's a different mental model than a project that you start, drive to completion, have a budget, etc."

The company now has only three or four project managers who oversee massive epics that cross many teams and usually involve customers. For example, a recent improvement needed intensive coordination with customers to upgrade 30,000 external connections. Although the work on the back end was reasonable, it took a long time. Project managers worked on this project for over eighteen months.

Business and Technology Synchronicity

As a product company, there is a dominant product management group. The chief product officer (CPO) owns the investment (about $500 million annually) for the product portfolio. Each product has its own profit and loss statement. All funding comes through the CPO.

In the scaling phase, the customer sales group served as the first line of defense to intake work. The group talked to product managers and coordinated backlog feature requests. The company had already completed a DevOps transformation, which included creating a single backlog for all work. This needed to be modified as it was found that a product manager tends to focus on new features, and as a result, can struggle to incorporate all "other" types of work, such as technical debt and incidents. To create a sustainable scale, the work types needed to be standardized. Part of the transition was to understand what all the different work types were and then create one backlog for all work. It was crucial that product managers on the business side became aware of and appreciate all the work—incidents, backlog, etc.—and that it was brought together and made visible. Much effort was expended to consolidate work management systems and to make sure that data was replicated between systems. Today, the product development teams control everything except for platform infrastructure and corporate applications, which reside with the CIO and corporate IT.

To sustain the new approach, another critical change was made. Every epic that comes in must have a business case to gain funding. Across the portfolio, work in progress is managed at the epic level. Service requests and incidents also come in. When the work was analyzed, it was found that work on incidents absorbed 40% of capacity. Even today, it is a continuing challenge to decide whether to do 40% less work or even 80% less work to eradicate those incidents. The advantage these days is that it is all managed through a single system, which was obviously not the case during scaling. Impact assessment is used to analyze each team's work capacity to ensure that teams are not overloaded. The same practices are used across the entire organization. This enables and is enforced by a lean portfolio group. The build toolchain is treated as a product and managed by a specific team. To underpin this way of working, engineering has become a core competence. This enterprise no longer uses contractors.

Product Taxonomy

As a product-focused organization by nature, taxonomy has always been standardized in all of the enterprise's market areas: workforce management, customer care and billing, order management, and messaging.

Architecture

The core architecture is SaaS-based with a multi-tenant back end. The earlier DevOps transformation drove most of the architectural changes, such as self-service and API-first design. In the scale and optimize phases, it was essential to avoid handoffs and create service-level agreements to maintain the integrity of the value stream.

The relationship with the enterprise team has evolved. Architecture patterns could be pushed to the platform team as it grew from a more traditional IT function. This helped the enterprise move away from ticketing systems and toward self-service. It also started to treat what it was doing as a product, as an investment, and as a service. Now, when the enterprise chooses to take on a new project, the enterprise performs an "impact assessment" that analyzes the project's architecture dependency. This happens every quarter. The enterprise plans which teams are involved, what platforms are required, etc. for each project.

In the previous phase, the ITIL was utilized. It was found that a lot of ITIL processes were very activity- and project-focused, not product-focused. Currently, the enterprise does not use ITIL-based change management for this. Change management is limited to production collision management.

Funding Model

The most essential change in the optimize phase is the funding model. Because the product model depends on teams being aligned to the products they build and run, it is crucial that as much of the spend as possible is also aligned to product. This means the product portfolio group looks at the overall investment plan. The decisions at the portfolio level are made annually. The chief product officer owns investment for the product portfolio and controls funding. Every year, decisions are made at the portfolio level about where to invest. There is an intake process based on epics. Each epic is funded independently.

Epics are presented every two weeks. Every Friday afternoon for ninety minutes the epics are proposed. All product groups “compete” in these meetings. There are politics, of course; since product managers control the funding, they have significant influence over how the epics are assigned. It is vital to convince product managers of the virtues of each epic. Some epics are rubber-stamped because they already align with a product manager’s strategy, while some are approved because the epic is mandatory. Everything passed gets visibility in that meeting.

The business case might not always be monetary; it could instead be employee satisfaction, customer experience, or reduction in hours. In this way, all products have P&Ls (profit and loss statements). Each product manager gets a slice of the funding pie. In this scenario, corporate IT is treated as a product. This has resulted in 90% of spend aligned to product.

One area that still needs refinement is charge-back for specific products. In theory, time is charged back to particular products, but it doesn’t work that way in reality. Because the cost structure looks exactly like the product layout, it would probably be better to get rid of chargebacks, but there’s still some resistance.

Culture and Leadership

As mentioned earlier, product/business alignment was already well understood in the enterprise. This meant that the business didn’t need to make many changes as project to product was scaled out. However, it was essential to extend the product focus to IT professionals who focus on the back-end systems. Enterprise IT lacked a dependable product management organization. Instead, the organization was run through projects and project managers, but that’s very different than product management. Project managers tend to be more mission-focused and tend to look less interested in investment ROI or spend. As the company started shifting to a product-centric model, they pivoted the focus toward product management, starting to treat each product group as the CEO of their product line.

Another change concerned operational service level objectives. In the previous phase, SLOs (service-level objectives) were viewed as a concern only for operations—operations owned SLAs/SLOs. These are now shared across product, engineering, and sales. Everyone shares the same objectives and goals. For example, if there is a target of 200 minutes downtime and it is exceeded, everyone feels the pain. Part of making all the work visible to all the stakeholders is that everyone needs to understand concepts like uptime and mean time to recovery. There was initial pushback from some stakeholders who felt that they didn't need to understand the role of operational aspects on overall cost, efforts, etc. That has been overcome.

Reflection

It was easy for this enterprise to holistically transition from a project to a product focus due to the existing product focus at its core. It was also advantageous that an expansive adoption of DevOps had already occurred.

Even so, the long-term sustainability of the project to product transition has required a change in the funding model so that 90% of all technology spend is aligned to business-recognized products. This requires a strong chief procurement officer, value stream thinking, and vulnerability to suggest and defend epics to business product owners in a competitive environment.

Other areas that needed to be overhauled to cement the long-term success of their project to product tranformation include calibrating project management capabilities and diluting the emphasis of ITIL support processes to collision management.

Conclusion

Moving traditional enterprises from project to product is a relatively new but significant movement occurring across all industries. While many enterprises are on the product transformation path, not many have fully realized the vision of a product-centric operating model. The community of leaders we interviewed are all considered thought leaders within the enterprise DevOps community, and the companies they work for tend to be early adopters in driving these changes. That said, only two or three of the fourteen companies we assessed are in the optimize stage of their transformation.

Because there are so many organizations that have yet to begin their product transformation journeys, we felt it was extremely important to capture real guidance for leaders who are trying to drive these changes within their own organizations. The seven domains of our framework (transformation implementation, business and technology synchronicity, product taxonomy, workforce and talent, funding model, architecture, and culture and talent) provide a meaningful structure for organizing and driving your transformation. Additionally, the added dimension of the transformation journey (incubate, scale, optimize) provides guidance on how to tackle these domains as you progress along your transformation. Finally, the constraints section of the framework provides useful lessons learned from the common failures that many of the companies faced along their own journeys.

While we believe this framework will be a useful tool to help leaders drive transformation, we do want to reinforce one key point as a closing thought. Frameworks don't transform organizations. Building high-performing teams and enabling them does. Don't blindly follow this framework and expect to achieve success. Look deeper into this guidance, think about your own organizational context, and apply the indicators to a strategy that best fits your environmental factors to accelerate change in your environment.

References

"Bimodal." Gartner IT Glossary. Accessed July 9, 2019. https://www.gartner.com/it-glossary/bimodal/.

Clanton, Ross, Carmen DeArdo, Mik Kersten, Alan Nance, Karen Person, and Jason Zubrick. *Moving from Project to Product: Modernizing Traditional Enterprise Operating Models*. Portland, OR: IT Revolution, 2018. https://itrevolution.com/book/moving-project-product/.

DeArdo, Carmen. "Project to Product: Practical Realities at a Large-Scale Enterprise," DevOps Enterprise 2018 London presentation, https://www.youtube.com/watch?v=3BQw5PTsCrU.

Denning, Steve. "Understanding Fake Agile." *Forbes*. May 23, 2019. https://www.forbes.com/sites/stevedenning/2019/05/23/understanding-fake-agile/#5ea731a54bbe.

"Devopsdays—Organizing Guide." Devopsdays.org. Accessed July 9, 2019. https://devopsdays.org/organizing/.

"The Flow Framework." *FlowFramework.org*. Accessed July 9, 2019. https://flowframework.org/.

Forsgren, Nicole, Jez Humble, and Gene Kim. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*. Portland, OR: IT Revolution, 2018. https://itrevolution.com/book/accelerate/.

Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. Portland, OR: IT Revolution, 2018. https://itrevolution.com/book/project-to-product/.

Kersten, Mik, Nanda Kumar, Daniele Romano, and Jason Zubrick. *Value Stream Architecture: Creating an Architecture to Connect the Dots in DevOps*. Portland, OR: IT Revolution, 2017. https://itrevolution.com/book/value-stream-architecture/.

Kissler, Courtney, Eric Passmore, Jeff Gallimore, Jeff Robke, Nicole Forsgren, Paula Thrasher, Pauly Comtois, Raphael Garcia, Rosalind Radcliffe, Ross Clanton, Scott Nasello, and Scott Willson. *Tactics for Leading Change: Evaluating What DevOps Patterns and Practices Would Work Best for Your Enterprise*. Portland, OR: IT Revolution, 2016. https://itrevolution.com/book/tactics-leading-change/.

"Overcoming Resistance to Change—Isn't It Obvious?" YouTube video, 6:13. "The World with Theory of Constraints." July 17, 2010. https://www.youtube.com/watch?v=hcz1aZ60k7w.

Reinersten, Donald. *The Principles of Product Development Flow: Second Generation Lean Product Development*. Redondo Beach, CA: Celeritas Publishing, 2009. https://www.amazon.ca/Principles-Product-Development-Flow-Generation/dp/1935401009.

Wiggins, Adam. "The Twelve-Factor App." 12factor.net. Last updated 2017. https://12factor.net/.

Notes

1. Gartner, “Bimodal”
2. Donald Reinersten, *The Principles of Product Development Flow*.
3. Mik Kersten, *Project to Product*.
4. Carmen DeArdo, “Project to Product.”
5: Mik Kersten, *Project to Product*.
6. Mik Kersten et al., *Value Stream Architecture*.
7. Donald Reinersten, *The Principles of Product Development Flow*.

A Special Thank You to Our Sponsor

Our mission for the Forum is to bring together technology leaders across many industries and facilitate a dialogue that solves problems and overcomes obstacles in the DevOps movement. For three days at this private event, we gather 50 of the best thinkers and doers in the DevOps space to tackle the community's toughest challenges. We ask these thought leaders to collaborate and generate a piece of guidance with their best solutions to the challenges.

We would like to thank all of our attendees and our friends at XebiaLabs for helping to make this year's Forum a huge success.

XL XebiaLabs
Enterprise DevOps