Outevolve

The Enterprise Operating Model for the Age of AI

Peter Beck Andreas Schliep

Table of contents

Preface

> [!note] Early Draft This book is actively evolving. Content may change. Feedback welcome.

Why Outevolve

AI will revolutionize your industry. AI will automate your workflows. AI will make your organization faster, leaner, smarter. That is the promise, and it is hard not to feel the pressure to act.

Here is another story: AI will displace your workforce, crash consumer spending, and trigger an economic crisis. Reports making this case have moved stock markets. This narrative creates fear and resistance, both of which paralyze organizations at the worst possible moment.

Neither story is likely the one we will tell in fifty years. AI is doing what every disruptive technology before it has done. It amplifies whatever structure is already there. And it forces a choice.

One path: use AI to accelerate what you already do. Deliver faster, produce more, automate harder. An efficient organization becomes even faster at delivering products and services customers no longer want. A bureaucratic organization generates more bureaucracy at machine speed. More reports nobody reads, more compliance rules nobody questions, more coordination layers nobody needs. This path looks like progress. It is not.

The other path: use AI as a catalyst to rethink what your enterprise does and how it is organized. Strip away complexity that no longer serves you. Redirect energy toward understanding customers, designing products, and making decisions under uncertainty.

This is the fork. Most enterprises will take the first path without realizing it.

But AI is only one force among many. And not the solution. Talent shortages are hitting every sector, and AI will increase demand for skilled and adaptive people, not reduce it. Regulatory burden is growing, not shrinking, and AI will even increase it by making each bureaucratic action cheaper to produce. Supply chains stretch across continents, fragile and exposed to geopolitical shifts, and AI will not make them less complex.

These challenges are not new individually. What is new is the pace. Changes that once took a decade now unfold in years. Multiple pressures hit simultaneously. And each one demands a different response: speed here, stability there, innovation in one area, efficiency in another.

The enterprises that thrive will be those that use this moment to strip away accumulated complexity and redirect energy toward what creates value: understanding customers, designing products, making decisions under uncertainty, collaborating across disciplines.

This requires more than deploying AI tools or launching another transformation program. It requires a different operating model. One designed not for stability or agility alone, but for evolution.

The real strategic question is not "Are we doing AI?" It is not "Are we agile?" It is: **How well does your enterprise navigate evolution compared to your competitors?**

This is what it means to outevolve.

You do not need another transformation program. You need an operating model that evolves with your company. One that is continuous, empirical, and grows one step at a time.

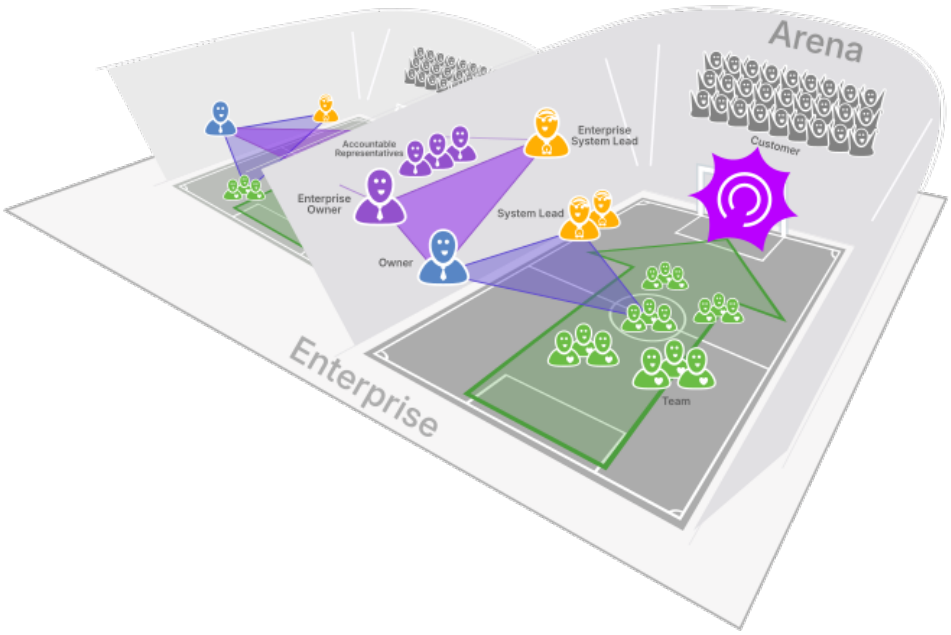
This book helps you build that operating model. We are Peter "Pit" Beck and Andreas "Andy" Schliep. Based on decades of experience in enterprise transformation, insights from the global agile and lean community, and research into strategy, organizational design, and complex systems, we developed AME3: the Adaptive Metaframework for Empirical Enterprise Evolution.

While AI defines the current moment, AME3 is not tied to it. The challenges enterprises face will not end when the next disruptive technology arrives. They will intensify. What you need is an operating model that works now and keeps evolving as the world around you changes.

What follows is a guide to playing a different game. A game where you do not just adapt. You outevolve.

> [!note] When we say AI in this book, we mean Generative AI, the technology behind large language models, image generators, and AI agents.

The New New Enterprise Game



In 1986, Hirotaka Takeuchi and Ikujiro Nonaka published The New New Product Development Game, an article that would profoundly influence how the world thinks about product development. Their insight was simple and powerful: small, cross-functional teams that work in overlapping phases outperform sequential, relay-race approaches. Speed, flexibility, and learning are built into the team, not managed from above.

This insight gave birth to the Agile movement. Frameworks like Scrum and eXtreme Programming helped software teams adapt to rapidly changing environments. Kanban and Lean brought the focus on flow, limiting work

in progress, and eliminating waste. These methods proved that empowered teams could deliver better results, faster. Countless organizations saw it work.

Then companies tried to scale these methods. With frameworks such as Scrum@Scale, SAFe, and LeSS, they pushed agility into larger parts of the organization. Yet the agile areas remained isolated, incompatible with the standard operation of the enterprise.

At best, agile teams were tolerated as eccentrics. At worst, they deepened the very silos they were meant to dissolve. More and more companies are now backing away from "Big Agile." It simply did not work for them.

The problem was never that agile methods lacked effectiveness. The problem was the absence of a compatible operating system for the entire enterprise. One that evolves with the company and integrates the principles that make agile frameworks powerful.

The Agile movement also had to learn that flexibility and adaptability alone are not what enterprises need. They must optimize for efficiency and stability as well. The real challenge is doing both at the same time. The answer lies in focusing on evolution, governed by empirical control.

It is time now to bring this game to the entire company. Welcome to **The New New Enterprise Game.** AME3 builds on the heritage of Agile and Lean to create an operating model for the evolving enterprise.

AI and the Principles of Evolution

Every major technology has promised to make our lives simpler. None of them did. The printing press, electricity, the internet, digitalization: each one was supposed to reduce complexity. Instead, we used every single one of them to build systems of ever higher complexity, pushing ourselves right back to the edge of what we can handle.

AI is no different. Understanding why tells us what will actually change for enterprises, and what will not.

Complexity Fills Available Capacity

Consider digitalization. Its promise was better, more reliable processes. Less paperwork, fewer errors, faster decisions. Look around any office today: people spend their days filling out spreadsheets, sending emails, managing workflows in tools they can barely keep track of. Digitalization did not reduce bureaucracy. We used it to create *more* rules, *more* processes, *more* administrative structures, because suddenly we *could* manage them.

This is not a failure of digitalization. It is a law of human systems.

We always operate at the maximum complexity our tools allow.

Before digitalization, we simply could not afford the level of bureaucratic control we have today. We would not have had the capacity to manage it. Digital tools raised the ceiling. We immediately moved in.

The same pattern repeats with AI. We will not use it to simplify our organizations. We will use it to build pharmaceutical research programs of unprecedented complexity, requiring entire Teams of specialists augmented by AI just to keep up with the science. We will use it to create financial products so intricate that no single human mind can fully comprehend them. We will use it to manage supply chains spanning continents with real-time optimization that would have been unthinkable five years ago.

We will use every bit of available AI capacity. Our data centers are already running at full throttle. When we build more, we will fill those too.

Life Creates Higher-Order Systems

Why does this happen? Because it is the fundamental principle of life itself.

Life, at its core, is a process that takes energy and uses it to create systems of higher order[1]. A cell takes chemical energy and builds a structured organism. A plant takes sunlight and creates complex molecular structures. An ecosystem takes the energy flowing through it and produces ever more intricate webs of interdependence.

[1] see: Higher-Order System, 197

Left alone, systems tend toward disorder, toward a uniform distribution of energy. Increasing entropy is a fundamental law of the universe: the second law of thermodynamics.

Life works against this tendency, locally. It takes energy and creates order, structure, complexity. In doing so, life actually increases the total entropy of its environment, releasing waste heat and disorder for every bit of order it builds. The local structure we see around us comes at the cost of greater disorder elsewhere. However, as long as energy flows in, life keeps building upward (see Life, Energy, and Entropy[2]).

Our economy is nothing other than an expression of this same principle. Our organizations, our products, our markets: they are all systems of higher order that we maintain through continuous investment of energy. Leadership energy, capital, human effort.

Teams, business units, and enterprises are how humans create higher-order systems. They are the social structures through which we channel energy into organized complexity. This is why maintaining a team costs energy: leadership, coordination, motivation. Stop investing, and the team dissolves. Thermodynamics.

AI Is Just the Next Evolutionary Step

The history of human progress is a history of co-evolution between humans and their tools. We create tools. The tools extend our reach. We use the extended reach to tackle harder problems. The harder problems require better tools. The cycle continues.

Fire, agriculture, writing, printing, steam engines, electricity, computing, the internet: each of these was an evolutionary step that fundamentally expanded what humans could achieve. Each time, the expansion was not toward simplicity but toward greater complexity. Each time, the social structures of collaboration adapted but did not disappear.

AI is the current step in this sequence. It is a powerful one, certainly. But it follows the same pattern. We are already using AI to develop AI, which is simply the logical continuation of this co-evolutionary process (see Open Ended Evolution[3]).

[2]see: What is the Purpose of Life?, 202
[3]see: Open Ended Evolution, 204

AI is not the end of evolution. It is the next chapter.

The question is not whether AI will change how we work. It will. The question is whether it will change the fundamental need for human collaboration. It will not.

Why Social Structures Persist

If you follow the argument so far, you might expect that AI will at least make teams smaller or less necessary. The evidence from every previous technological revolution says otherwise.

Consider the discussion around team size and structure. Some argue that AI will dissolve traditional teams into fluid, dynamic configurations: pods, wings, pairs with AI assistants, dynamic reteaming within larger groups. These structures are not new. In well-functioning scaled organizations like LeSS[4], you already see groups of 50 to 60 people who know each other well enough to reconfigure their team structure from Match to Match (or Sprint to Sprint) as needed.

This is why the concept of a "team" is deliberately broad in AME3[5]. There is no fixed number. A hospital ward operating in shifts is a team, even though its members rarely see each other all at once. A software development group of seven is a team. An Arena of 200 people can function as a cohesive unit with dynamic sub-structures, containing many teams of different sizes and lifespans within it.

Think of a football squad. The team is not just the eleven players on the pitch. It includes substitutes who never play a single match but are essential to the squad's readiness. The club forms an Arena in terms of AME3. It includes the coaching staff, the marketing team, the fan and community liaison group. All of these Teams work together within one stable unit, each contributing to the club's success in their own way, toward a common Ambition and shared Goals. The club itself is the Enterprise. A large one has multiple Arenas, focused on the women's league, the junior program, the premier league.

The social pattern of the team predates every technology we have ever invented. It will outlast AI too.

[4] see: LeSS, 201
[5] see: AME3, 185

What changes is what teams take responsibility for. As described in The AI-Enhanced Team of the Future[6], teams are evolving from component-focused groups to full-business teams that own end-to-end value chains. AI accelerates this evolution. It does not replace the team itself.

New Products Require Reorganization

There is one dimension where AI's impact goes beyond previous technological shifts: the sheer speed at which new products and services can emerge.

When AI enables a team to build, test, and deploy a new product in weeks rather than months, the bottleneck shifts from development to organizational adaptation. Your enterprise must be able to absorb new products into its portfolio, spin up new Arenas, restructure value chains, and reallocate resources, all within the strategic cadence of months, not years.

This is where Evolution Focus[7] becomes critical. Every advancement along the evolution path requires reorganization. This is not dramatic in an AME3 organization. It is standard practice. The flexibility to reorganize is built into the system through Empirical Control[8] and the lean governance structures of the Enterprise.

But make no mistake: the pace of reorganization will increase. Enterprises that treat organizational structure as fixed will fall behind those that treat it as a living system, one that evolves alongside its products and markets (see also Developing a Strategy for the GenAI Era[9]).

The Principle

Life creates systems of ever higher order as long as energy is available. Humans always push their systems to the maximum complexity their tools allow. AI is the latest and most powerful tool in this sequence, but it follows the same evolutionary pattern as every tool before it.

[6] see: The AI-Enhanced Team of the Future, 117
[7] see: Evolution Focus, 37
[8] see: Empirical Control, 20
[9] see: Developing a Strategy for the GenAI Era, 106

This means we will keep reinventing our products and services, and keep reorganizing our organizations to deliver them. AI accelerates this cycle, just as every major invention before it did. We will simply reinvent and reorganize faster.

What remains are the fundamental social structures humans need to thrive, to communicate, to think, to organize together toward a shared purpose.

As long as we have energy, we will build. And as long as we build, we will build together.

AME3 Stands Proudly on the Shoulders of Giants

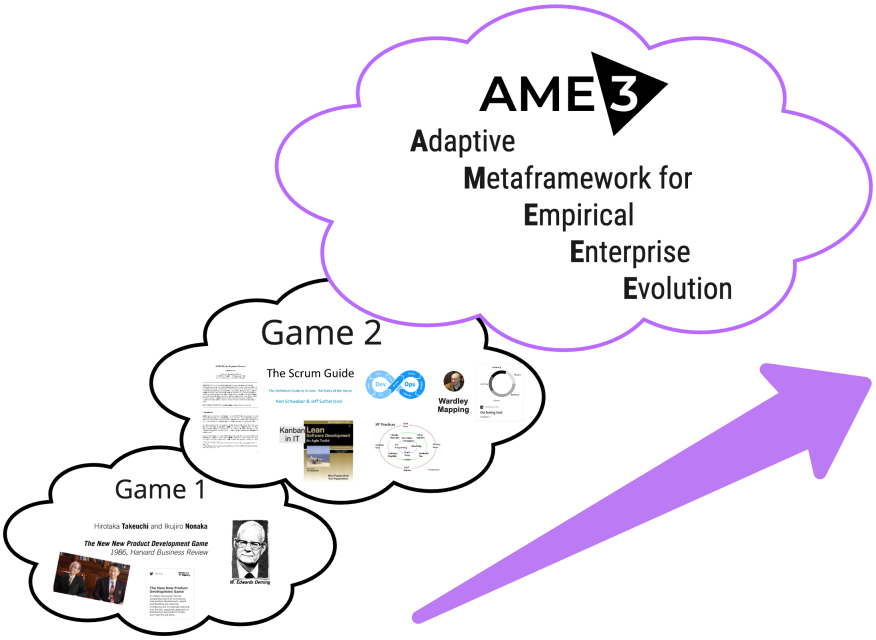


Figure 1.: Evolution of methods and frameworks for product development, organizational design, and strategy

AME3 is built upon the foundational work of numerous methodologies and frameworks in agile, lean, and enterprise transformation. The visual above

shows how AME3 synthesizes insights from established approaches including Scrum, Kanban, Lean, LeSS, and many others to create a comprehensive enterprise evolution framework. This itself is evolution, a fundamental concept behind AME3 and one of the core Strategic Doctrines an enterprise should focus on.

Rather than replacing these proven methods, AME3 integrates their core principles while addressing the unique challenges of large-scale organizational transformation. From Scrum comes the power of self-organizing teams, clear leadership, and the focus on done work. From Kanban and Lean, the focus on flow, limiting work in progress, and eliminating waste. From Large Scale Scrum, the lesson on aligning multiple teams toward shared goals. From Wardley Mapping, the insight that agility itself should not be the primary optimization goal. Instead, an enterprise needs to focus on the evolution of its products, services, and organization.

AME3 weaves these influences together into a coherent system designed for entire enterprises, not just individual teams. The AME3 Rules alone are not enough to enhance agility and results. Enterprises should adopt complementary methods and frameworks such as those mentioned above. The choice should align with the evolutionary stages of the product and organization, and with the particular Ambitions being pursued.

Some of these frameworks and methods may have different rules and recommendations. The combined use still outweighs the inconsistencies. The Rules of AME3 support easy mapping and integration with other frameworks, enhancing overall results.

Unlike prescriptive scaling frameworks, AME3 gives organizations the flexibility to steadily adapt their structures and processes to meet the demands of a competitive environment. The agile frameworks and methods recommended within AME3 may occasionally conflict in detail. In cases of discrepancy, Empirical Control is invaluable. This again is a Strategic Doctrine in AME3.

Who This Book Is For

This book is for anyone responsible for how an enterprise evolves. You may lead one, advise one, build its products and services, or manage its projects. Here is how AME3 meets you where you are.

You lead the enterprise. You are a CEO, founder, or managing director facing transformation pressure from multiple directions: technology shifts, talent shortages, rising costs, and boards asking hard questions. You need a framework that connects strategy to execution without requiring a multi-year consulting engagement. AME3 gives you an operating system that evolves with your company, starting from where you are today.

You sit on the board. You oversee strategy and governance across one or more companies. Management teams present ambitious transformation plans, but you lack the tools to evaluate whether they will work. AME3 gives you a clear framework for asking the right questions and distinguishing real progress from activity.

You lead technology. You are a CTO or VP of Engineering caught between legacy systems, AI pressure, and teams that cannot move fast enough. Everyone expects you to deliver faster, but the organization was not designed for speed. AME3 shows you how to structure teams for faster delivery and align technology investments with business strategy.

You run a business unit. You lead a product line, division, or department inside a larger organization. You see the opportunity, but corporate processes slow every decision. Agile methods helped your teams improve, yet the surrounding organization did not change. AME3 gives you a path to create real autonomy within corporate constraints and demonstrate measurable value.

You are scaling a company. You built something that works, and demand is growing faster than your organization can handle. What got you here will not get you there. AME3 helps you add structure without losing the speed and focus that made you successful.

You are preparing for a transaction. A spin-off, buy-out, or IPO is on the horizon. You need to demonstrate operational maturity and build an organization that can stand on its own. AME3 provides the structure buyers and investors look for: clear governance, measurable outcomes, and an organization designed to operate independently.

You advise enterprises. You are a management consultant, and your clients ask for help navigating complexity. Existing frameworks feel too rigid for the diversity of situations you encounter. AME3 expands your toolkit with a meta-framework that composes proven methods rather than prescribing a single approach.

You coach teams and organizations. You are an agile coach, Scrum Master, or organizational development professional. You have seen agile methods work at the team level but struggle to connect that success to enterprise strategy. AME3 bridges this gap. It gives you a language and structure to extend agile principles beyond teams, into the operating model of the entire enterprise.

You manage projects and portfolios. You are a project manager, program manager, or PMO lead responsible for delivering results across teams and departments. Traditional project governance no longer fits the pace of change. AME3 shows how portfolio management, goal-setting, and cross-team coordination work in a product-oriented enterprise. It gives you a clear path from project thinking to continuous value delivery.

You shape the people side of transformation. You lead HR, talent development, or organizational design. Every transformation ultimately depends on how people work together. AME3 provides a framework where organizational structure follows strategy, roles carry clear accountability, and teams are designed around products rather than functions.

All these paths lead into this book. How to Read This Book will help you find the right starting point.

How to Read This Book

You do not need to read this book from cover to cover. Each part stands on its own. Pick the one that matches where you are right now.

Part One: System

The System is the operational model at the heart of AME3. It defines how your enterprise leads, governs, and evolves.

The Strategy answers what to evolve and where to invest. Three strategic doctrines guide this. Empirical Control structures your learning through Anticipate-Advance-Assess cycles at multiple timescales. Overall Optimization ensures that local improvements benefit the entire enterprise, not just the unit making them. Evolution Focus maps your products along their evolution path, from genesis to commodity, so you invest where it matters most.

The Leadership System introduces three balanced leadership functions: the Owner, the System Lead, and the Team. Together they form a triangle of forces that keeps the organization stable yet adaptive. You will learn why leadership is a system design challenge, not a question of individual traits. Each function carries clear responsibilities: Owners set direction and balance opportunity against risk. System Leads develop people and improve the work system. Teams manage and execute work while driving quality and customer satisfaction.

The Rules establish a minimal yet sufficient framework for enterprise-wide collaboration. They operate on two levels: the Arena, where Teams create products and services, and the Enterprise, where strategic decisions shape the portfolio. You will find leadership functions, artifacts, and constraints at both levels, just enough structure to enable integration with proven methods like Scrum, LeSS, and Kanban, without prescribing how to do the work.

Part Two: Playbook

The Playbook is your step-by-step guide for getting started. It turns the System into action without requiring a large-scale transformation.

The first entry point is the enterprise. You start the game at the enterprise level by defining the Leadership functions: the Enterprise Owner, Accountable Representatives, and Enterprise System Lead. You adopt the Rules as your shared operating agreement, analyze your product portfolio for innovation potential, and build an Enterprise Backlog that orders all strategic Goals. Then you launch the first Tournament, the enterprise-level cycle where strategy meets reality and adjusts based on evidence. Finally, you define your first Arena and its Ambition.

The second entry point is the Arena. You start the game in an Arena from one of three starting situations: a brand-new product independent from existing processes, an existing work system that already uses agile and lean practices, or an established product that has not yet adopted them. Each path provides detailed steps, from setting up a Leadership System with Owner, System Lead, and Teams to creating a single Backlog and starting the first Match.

You do not need to reorganize everything at once. Employees move to Arenas as they are needed. The rest of the organization continues operating

as before. The enterprise evolves one Arena at a time, guided by strategic Goals and governed by the Tournament cycle.

Part Three: Interplay

Interplay connects the System to the real world. It shows how organizations apply AME3 to the challenges they face today.

The AI section explores how generative AI is reshaping business fundamentals. You will learn why AI will not fix your bureaucracy, how to assess your strategic terrain using Wardley Mapping, develop a strategy for the GenAI era, and how Teams are evolving from component teams to full-stack teams to AI-enhanced Teams that own end-to-end business outcomes.

The Methods section curates practical guides for selecting, adapting, and combining proven frameworks. You will find a detailed analysis of what makes framework adoption succeed or fail, and why AME3 works as a meta-framework that composes rather than prescribes.

You will learn how to use methods like Scrum, LeSS, Wardley Mapping, and Estuarine Mapping within AME3 for the benefit of your enterprise, your products, and your services. Other articles cover the transition from project to product thinking, enterprise Goal visualization, and practical tips for planning and estimation at every level of the organization.

Each article and method stands alone. Together they show how the System comes alive in practice.

> [!note] A note on our voices Where you read “I,” it is me, Peter Beck, or simply Pit, sharing a story, my beliefs and ideas. That is not because I love storytelling more than Andreas Schliep (or simply Andy). It is because I am the main author and editor of this book. Andy, once called the living Agile Wikipedia, contributes with background chapters on models and methods. His deep knowledge and his habit of challenging my ideas shaped every concept. Without his contribution, this book would not exist. If Andy shares a story, I will make sure you notice it. This book is a creative co-work, fueled by our different personalities. We hope you enjoy reading it as much as we enjoyed writing it.

Part I.

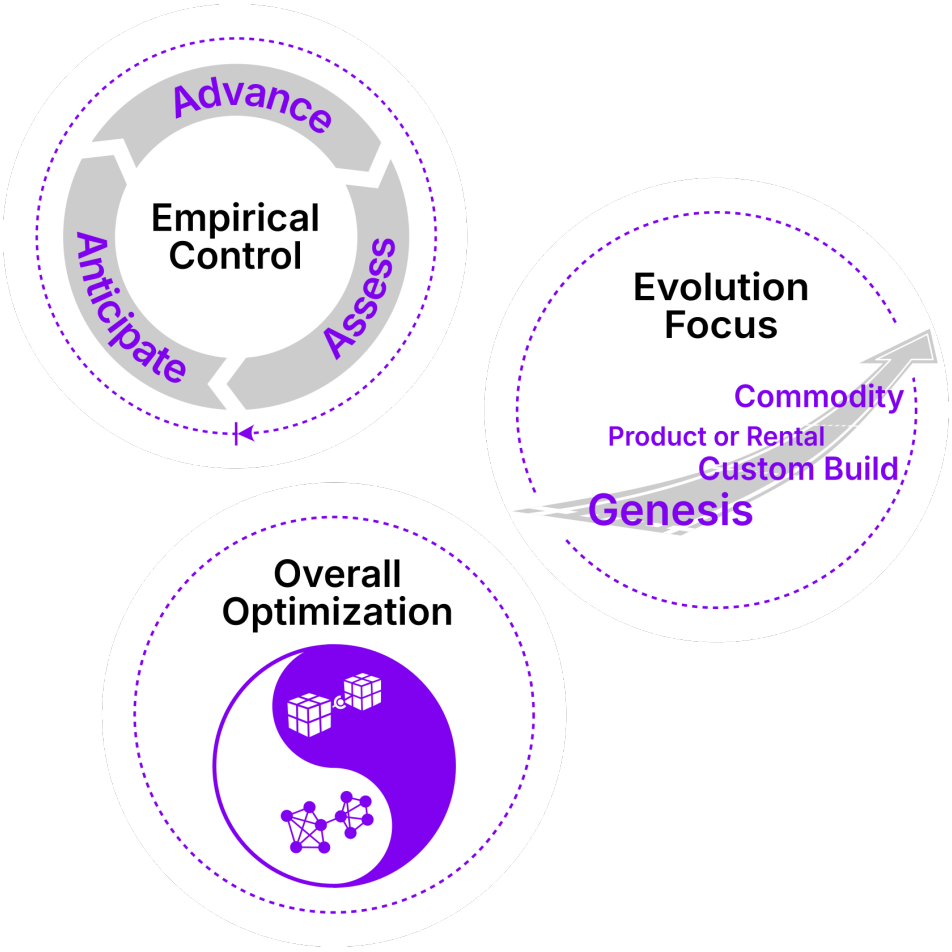
System

AI is reshaping markets, business models, and the way enterprises create value. But it is not the only force. Shifting customer expectations, regulatory pressure, and technological evolution have always demanded that organizations adapt. What AI adds is speed and uncertainty at a scale most operating models were never built to handle.

An enterprise needs a system that is robust enough to direct these forces and universal enough to work regardless of what comes next. AME3 provides that system. It is a foundation for building and evolving an operating model unique to your enterprise. One that balances stability with adaptability while continuously optimizing for the services and products that matter most.

The System comprises three interconnected components. Strategy guides organizational evolution through three strategic doctrines. Leadership defines how Owners, System Leads, and Teams collaborate as a self-regulating triangle of forces. Rules establish a common language and foundational practices for teams and leaders alike.

1. Strategy



Strategy in AME3[1] is not a plan. As Roger Martin[2] puts it, planning is about activities within your control, while strategy requires making choices

[1] see: AME3, 185
[2] see: A Plan Is Not a Strategy, 183

about where to play and how to win in the face of uncertainty. A plan tells people what to do. A strategy tells them how to think when the plan runs out. It does this through doctrines.

In military strategy, a doctrine sets out the fundamental beliefs, priorities, and methods that shape policies, operations, and the use of power over time. It provides a framework for tactics. The concept originates from US Air Force doctrine (AFDD-1), where it describes "officially sanctioned beliefs and warfighting principles that guide the proper use of forces."

> A **Strategic Doctrine** provides a framework for tactics, setting out the fundamental beliefs, priorities, and methods that shape policies, operations, and the use of power over time.

Simon Wardley[3] brought the concept into business strategy, defining doctrine as "basic universal principles applicable to all industries regardless of the landscape and its context." In AME3, we take this idea further and establish three strategic doctrines that every enterprise must adopt from the start.

Empirical Control[4] structures learning through Anticipate-Advance-Assess loops at multiple timescales. Overall Optimization[5] ensures that local improvements benefit the entire enterprise, not just the unit making them. Evolution Focus[6] maps products along their evolution path so the enterprise invests where it matters most.

Empirical Control

Expecting the Unexpected

During my studies in electrical and information technologies in the late 90s, I (Pit) had the opportunity to work as a student assistant with a team of engineers and physicists at IBM. Their main task was the production of new hard-disk generations. More specifically, they were responsible for the disk coating process, which involved creating layers of metals and carbon.

[3] see: Wardley Mapping, 217
[4] see: Empirical Control, 20
[5] see: Overall Optimization, 29
[6] see: Evolution Focus, 37

These layers, measured in Ångström (a unit roughly equivalent to the diameter of an atom), were created using a well-established process known as magnetron sputtering. This process utilized a sequence of ultra-high vacuum chambers with a controlled plasma in each chamber. The entire product was manufactured in a clean room fabrication facility, roughly the size of two football fields.

As fascinated as I was by the ongoing engineering in this high-tech environment, I was equally surprised by how the team operated when bringing a new disk into production:

First, the team formulated a hypothesis outlining the parameters necessary to achieve the desired properties of the final disk. These parameters encompassed a wide array of variables for the plasma, such as voltage, gasses introduced into the vacuum chamber, the material itself and many more I never fully understood. The hypothesis was grounded in theoretical models, laboratory experiments, and knowledge acquired from past products.

After producing a larger batch of disk under production conditions, they started measuring the characteristics of the disks. Most likely they failed to achieve them, but the data gave the team hints where to adapt the parameters, which led them to plan the execution of the next cycle.

At times, it took the team weeks and numerous cycles to achieve their desired results. During this pursuit, they had to eliminate common issues, such as a stray hair or a fingerprint fragment in one of the chambers.

Development is the art of expecting the unexpected.

The team also found themselves needing to reevaluate their initial assumptions. For instance, they came across an issue involving minor but quantifiable deformations on the disks. Initially, the team hypothesized that these deformations were due to the plasma's extreme temperature, a seemingly logical assumption. However, the real culprit was the forces produced by high acceleration when the disks were inserted into the production machine. As a result, they needed to optimize the loading robot's movement.

Empirical: *Based on what is experienced or seen rather than on theory*

Years later, while working as a programmer, I discovered the term for this meta-process: *Empirical Process Control*. This is how Ken Schwaber de-

scribed the cycle in Scrum[7], known as a Sprint[8]. In his work, he explained that he had learned this concept from professionals in the chemical industry. (Process Dynamics, Modeling, and Control[9]).

At this time, I was already familiar with this type of work from eXtreme Programming[10], which operates on shorter loops. Initially, you define your expectations for the code change by writing a test case that naturally fails (marked as red). Then, you write the code to pass the test case (turning it green). Finally, you analyze the code and begin to refactor it, ensuring it still passes all existing tests. This completes one cycle, and readies you to start the next one.

In 2004, Andy and I experienced Scrum for the first time as a developer. We devloped a new web comuication platofrm for, at this time, largest web-protal in germany.

Sprints offered us developers an expanded cycle that facilitated our journey from a mere business concept to done functionality. What we later learned was that the highest form of "Done" signifies that our users and customers have actively started utilizing the new features. Only then do we gain the invaluable insights that serve as the definitive test for our business hypotheses.

But individual Teams running Sprints was only the beginning. In 2007, we helped Bwin scale Scrum across the entire development and IT department. When all Teams adopted the same rhythm, organizational noise and chaos dropped significantly. The structure we built there was later described almost exactly by Large Scale Scrum (LeSS[11]), a pattern for a large number of Teams working on one product.

Looking back, these Empirical Control Loops were not just optimizing the process to develop software. They controlled the entire result we were creating: our product, or in AME3 terms, the Arena Product.

Scrum, XP, and Agile did not invent this. Empiricism[12] itself, as both a scientific and philosophical approach, dates back to the 17th century. In

[7] see: Scrum, 210
[8] see: Sprint, 213
[9] see: Process Dynamics, Modeling, and Control, 206
[10] see: eXtreme Programming, 219
[11] see: LeSS, 201
[12] see: Empiricism, 191

fact, you can find these loops in models, methods, and organizations across every domain:

Examples of Empirical Control Loops

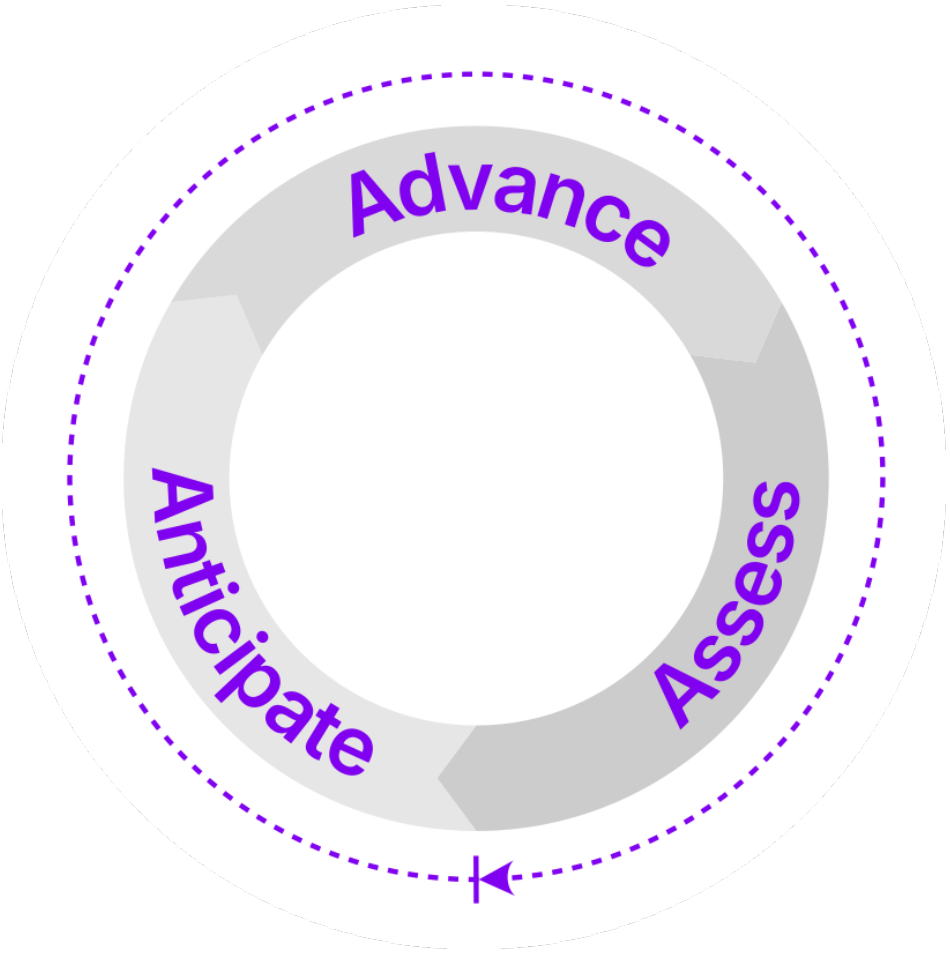
- OODA Loop[13]: Observe, Orient, Decide, Act
- PDCA Cycle[14]: Plan, Do, Check, Act
- Wardley Mapping: Leadership, Purpose, Landscape, Climate, Doctrine
- eXtreme Programming: Red, green, refactor
- Lean Start-up[15]: Build-Measure-Learn
- Process for the complex domain in the Cynefin Framework[16]: probe-sense-respond
- Scrum: Transparency, Inspecting, Adapting, in combination with time-boxing and deliver early and frequently. Manifested in the Sprint and the Daily Scrum[17]

While these loops differ in terminology and steps, they all share the same underlying pattern. AME3 distills this into three phases.

Anticipate, Advance, Assess

In AME3, there are two defined empirical loops. The first, at the tactical level, is the Match within an Arena. The second, at the strategic level, is the Tournament encompassing the entire Enterprise. They are structured into three phases:

[13] see: OODA Loop, 203
[14] see: PDCA Cycle, 205
[15] see: Build-Measure-Learn, 187
[16] see: Cynefin, 189
[17] see: Daily Scrum, 189



Anticipate

Anticipate refers to the phase where the organization or work system formulates hypotheses and sets expectations for the upcoming cycle. This involves reconsidering strategic doctrines, constraints, and defining the parameters and conditions necessary to achieve the desired outcomes. Decision-making and planning are also integral components of this phase.

Advance

The *Advance* phase involves executing the planned actions and improvements by adhering to the strategy doctrines, constraints, tactics, and plans outlined during the Anticipate phase. This is where the organization actively engages in the process, whether it's developing a new product, conducting experiments, or implementing changes. During this phase, the organization gathers data and observes the outcomes, making real-time adjustments as needed to stay aligned with their goals and improvements.

Assess

The *Assess* phase focuses on evaluating the results of the actions taken during the Advance phase. This involves analyzing the data collected, comparing the outcomes against the initial hypotheses and expectations set during the Anticipate phase, and identifying any discrepancies or areas for improvement. The insights gained during this phase inform the next iteration of the AAA-Loop, allowing the organization to refine its strategies and approaches continuously.

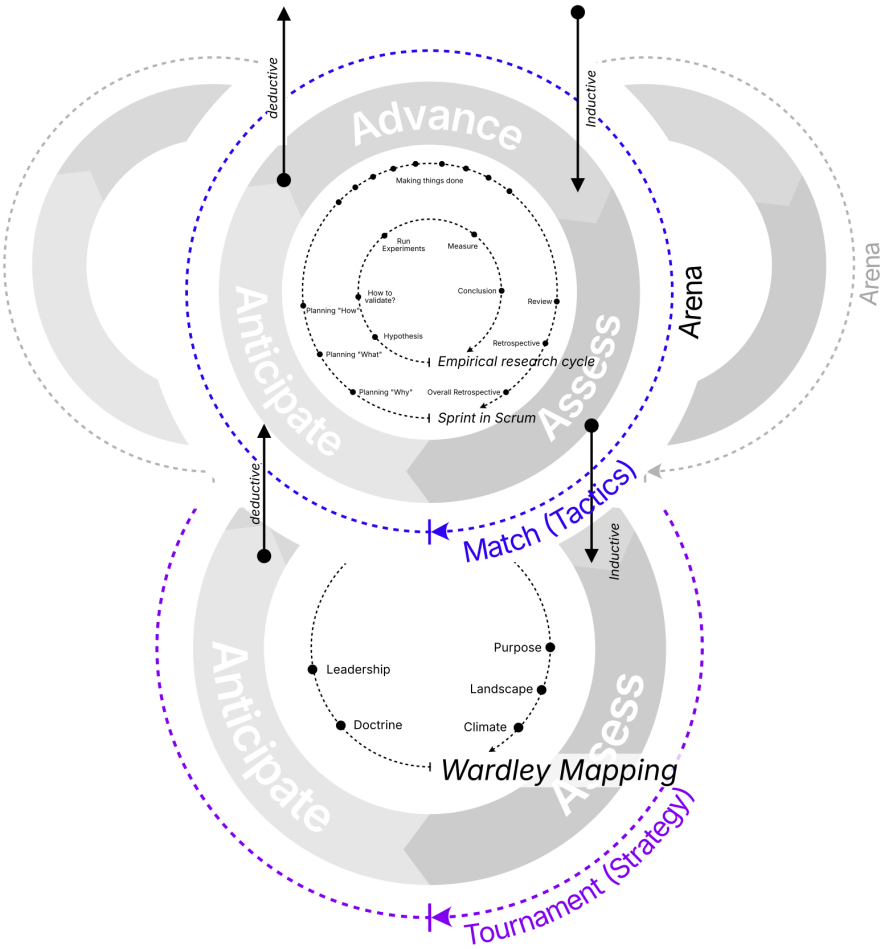
The phases are not isolated from each other and can have some overlap. For instance, the IBM team collected a significant amount of data during the Advance phase and drew conclusions from it instantly, which can be considered as the Assess phase. However, in a meeting, they summarized these insights and anticipated future experiments before advancing to the next cycle.

Strategy and Tactics

The Rules of AME3 help implement the AAA-Loops and therefore the Empirical Control doctrine. The organization and facilitation of the Tournament and Match are determined by the System Leads and the particular methods and frameworks they elect to use. However, the shared rules ensure that the loops are interlocked.

As an example, the following image illustrates how the Tournament and Match integrate three frameworks. On the strategic level, the Tournament incorporates Wardley Mapping. On the tactical level, the Match integrates

Scrum and a research cycle similar to the one used by the IBM team. However, each Arena likely has its own implementation for a Match, utilizing different methods and frameworks.



On the strategic level, the loop of the Tournament does not have its own Advance phase. This phase consists of the combined Matches of all Arenas and the remaining organization that does not operate within the AME3 framework.

Only Teams and individuals who actively collaborate with customers or solve problems for them will advance.

The Perfect Length of a Loop

It's clear that the shorter the AAA-Loop, the quicker the improvement and optimization process. However, certain constraints may necessitate a longer period:

- Technological complexity
- Market complexity
- Organizational complexity
- Social constraints

Often, we tend to select the time based on the constraints at hand. For instance, in the development of a new drug, the duration of a clinical test seems to dictate the maximal time for the loop. However, this approach has a drawback: we accept our assumption that we could not accomplish the goal more quickly. As an example, the development of the COVID-19 vaccine by BioNTech demonstrated that perceived constraints were actually unchallenged assumptions, as evidenced by their rapid vaccine development initiative[18].

By setting a fixed timebox, we flip the problem on its head:

- We challenge ourselves to break down the problem into smaller, more manageable parts from which we can learn.
- We adopt a divide and conquer strategy to tackle the problem.
- We concentrate on fewer problems and resolve them in less time, providing us with better data faster for future decisions.
- We establish a rhythm that reduces noise and chaos in the organization.

Designing the Loop in Loop in Loop

So, designing an AAA-Loop seems to be a trade-off. In larger organizations, these loops need to interconnect and build upon each other. It's clear that combining numerous loops can result in confusion and disarray. Fortunately, the Agile community's decades of experience have shown that three interlocked loops are sufficient for most small and medium enterprises:

[18] see: Projekt Lightspeed, 206

1. An hour to a day long AAA-Loop for tasks handled by individuals and the Team, led by
2. a week to a month long AAA-Loop for end-to-end improvements involving one or multiple Teams, led by
3. a quarter to a year long AAA-Loop for the overarching Goals and Ambitions of the Enterprise.

Several frameworks and methods effectively address one or two loops. Others are useful for supporting the loops, but do not define a loop on their own.

The challenge often lies in the fact that different parts of the organization may not operate in sync, which causes issues in communication and collaboration. Moreover, these organizational parts may not follow the same doctrine of empirical control, which is crucial for planning common Goals and integrating data from the lower AAA-Loops.

By elevating Empirical Control to an overarching strategic doctrine, the Enterprise ensures that various subsystems within the organization collaborate effectively within a generic structure.

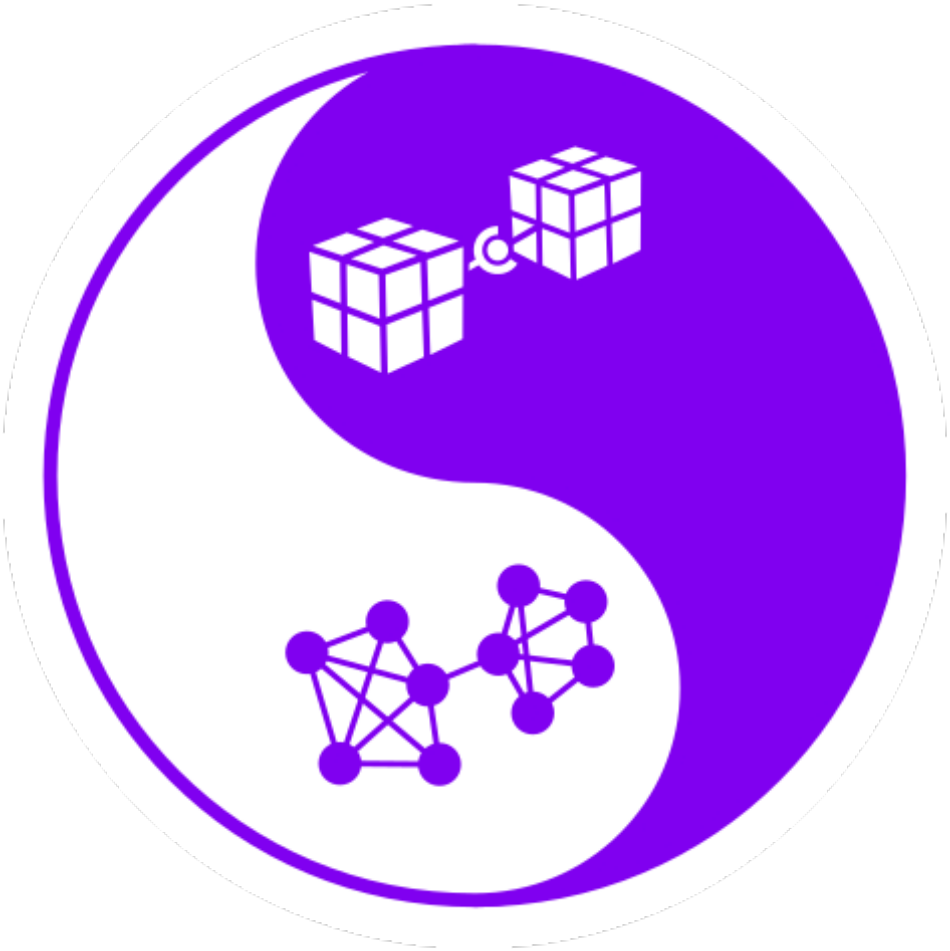
AME3 aims to maintain these AAA-Loops as flexible as possible, allowing different organizational parts to select the specific processes needed for their situation while providing enough structure to ensure subsystem integration.

Next Level

Empirical Control is prevalent in various models, methods, and organizations. It is a fundamental component of effective product and service development and should, therefore, be the first strategic doctrine of an Enterprise.

In the context of AME3, these cycles are embodied in the Tournament and Match. But running empirical loops is not enough. The loops must serve the right scope. When different parts of an organization each optimize for their own results, the enterprise works against itself. The second strategic doctrine, Overall Optimization, ensures that every improvement benefits the whole.

Overall Optimization



The Suboptimization Trap

In 2007, I (Pit) joined Bwin in Vienna as a Scrum Master, eager to apply what I had learned about agile development. My disappointment came quickly when working with my first team. Despite everyone's best efforts, the team never delivered what was planned by the end of each Sprint.

Management blamed the team for lacking commitment. But the real problem was different.

Each team member had received individual performance goals from their respective managers. These personal objectives pulled people in different directions. The developers optimized for their individual targets, not for the team's shared outcome. What looked like a commitment problem was actually a suboptimization problem.

When everyone optimizes for their own goals, nobody optimizes for the whole.

This pattern repeats across enterprises of all sizes. Departments optimize their budgets. Teams optimize their velocity. Individuals optimize their performance reviews. Each local optimization makes perfect sense in isolation. Together, they create an enterprise that works against itself.

Overall Optimization means every improvement must be evaluated by its effect on the Enterprise Product, not just the unit making the change.

When the Whole Enterprise Aligned

A fortunate circumstance allowed me to experience what happens when an enterprise truly optimizes for the whole. Bwin's board decided to launch a loyalty program called b'inside with the highest priority. This decision overrode all individual targets and departmental goals.

Team members were not assigned. They were asked if they wanted to work on this product idea. True commitment emerged because people chose to dedicate themselves to a shared Goal.

When obstacles arose, I could simply remind everyone of the overarching objective. The "Project #1" priority card cut through organizational friction. Within the first Sprint, the team delivered four complete features with automated testing, unprecedented in that organization.

After four Sprints, the team reached eight times the velocity. But velocity was not the real achievement. The team tackled fundamental obstacles that no one had dared to address before: replacing legacy systems, restructuring the entire infrastructure, and removing bottlenecks that had slowed the organization.

True commitment arises only when people decide for themselves to pursue a shared Goal.

By 2009, Bwin had scaled to nearly 20 teams delivering continuously every four weeks. This capability was rare for that era. The difference was not better processes or smarter individuals. The difference was alignment around a shared outcome.

The Challenge of Scaling

As enterprises grow, communication becomes increasingly complex. Without structure, communication effort grows quadratically. Ten people require 45 communication channels. One hundred people require 4,950. This explains why growing organizations feel chaotic despite everyone working harder.

Teams offer a solution by compartmentalizing communication. Internal team communication stays manageable while inter-team communication handles coordination. However, this structure has a critical threshold.

The team-based approach breaks down when Teams must coordinate constantly with each other. At that point, the benefits of team structure disappear.

No organizational approach scales linearly. The question is how to manage the inevitable complexity.

You cannot simply add more Teams and expect proportional output. Dependencies between Teams create coordination overhead that can consume all productivity gains.

Conway's Law and the Architecture of Work

In 1967, Melvin E. Conway observed that organizations produce designs mirroring their communication structures. This insight, now known as Conway's law[19], reveals why reorganizations often fail to change product architecture.

> "Organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations." —Melvin Conway

[19] see: Conway's Law, 188

Product architecture and organizational structure are two sides of the same coin. You cannot change one without changing the other. An enterprise optimizing its product must simultaneously optimize its organization.

But Conway added a critical insight that many overlook:

> "Because the design which occurs first is almost never the best possible, the prevailing system concept may need to change. Therefore, flexibility of organization is important to effective design." —Melvin Conway

At Bwin, we (Andy had joined as Scrum Master in the meantime too) experienced this directly. A Swedish component team created a dependency bottleneck that blocked progress. The solution was not better processes or more meetings. The solution was co-location, bringing the Swedish developers to Vienna. Once communication patterns changed, the architectural problems resolved themselves.

The lesson: treat organizational design and technical architecture as one problem, not two. Optimize both for flexibility, the ability to adapt when the prevailing system concept needs to change. And they will change.

The De-Scaling Cycle

Scaling reveals dependencies. These dependencies appear in two interconnected forms: between Teams and organizational units, and within the structure of products and services. Conway's Law tells us these are the same problem viewed from different angles. Organizational dependencies mirror product and service dependencies.

Managing both requires a systematic approach. In AME3, we call this the De-Scaling Cycle, a continuous practice of reducing complexity across organization and product structure:

1. **Identify Dependencies.** Not all dependencies are visible from the start. New ones emerge as products and services evolve. Look for dependencies between Teams and organizational units that force constant coordination. Look for coupling in product and service structures that prevents independent work. These are the same problem in different forms.

2. **Manage Communication.** The immediate response is to manage work and communication along the dependency. For Teams and organizational units, this means taking responsibility for coordination. For product and service structures, this means establishing clear interfaces. For time-limited problems, management is often sufficient.
3. **Reduce Dependencies.** Sustainable improvement comes from eliminating dependencies, not just managing them. This requires simultaneous changes to organization and product structure. Restructure Teams along stable boundaries. Decouple product and service components. Because these dependencies mirror each other, addressing one often reveals solutions for the other.
4. **Repeat.** Each improvement reveals previously hidden dependencies. New dependencies emerge as products and services expand. The cycle never ends—it becomes part of how the enterprise operates.

Measuring What Matters

After Bwin adopted Scrum across the entire product development organization, I observed a curious phenomenon. Management introduced a KPI for Scrum Masters: increase commitment reliability. The metric was simple, the ratio of committed story points at the start of a Sprint to the story points actually delivered.

Guess what happened? Teams did not get faster. They did not get slower either. They delivered with exactly the same velocity, Sprint after Sprint. In a complex environment, this is odd. Velocity naturally fluctuates as teams tackle different challenges, experiment, and learn.

The root cause was subtle. Scrum Masters, optimizing for their KPI, held teams back from challenging themselves. They encouraged conservative commitments to keep the ratio stable. The metric was perfectly met, and innovation quietly suffocated.

Management's intention was understandable. They wanted reliability and planability on one hand, and more features and innovation on the other. But these goals are a trade-off. The KPI optimized for one side only. And they got exactly what they measured. **What you measure is what you get.**

This pattern repeats everywhere. When departments measure their own efficiency, they optimize for departmental metrics. When Teams measure

their velocity, they optimize for story points. These local measurements drive local optimizations that may harm the whole.

The Agile Manifesto shifted measurement from completed activities to working software. AME3 extends this principle further: the measure of success is the Enterprise Product, not the output of individual units.

Goal and Ambition

The Enterprise Product represents all products and services offered by the enterprise. It combines Arena Products from all Arenas with results from units not yet operating within AME3. Avoiding suboptimization while slicing products and services into manageable units is a constant tension. The system design of an Arena should be optimized for low dependency, so that Teams can deliver value independently without harming the whole.

AME3 addresses this tension through two complementary artifacts: the Ambition and the Goal.

The Ambition defines the purpose, expected successes, and constraints for each Arena Product, including financial limitations. It justifies the existence and operations of an Arena. The Owner is responsible for leading the Arena Product towards achieving the Ambition. If the Ambition is at risk, the Owner must take appropriate actions, which may include terminating the Arena Product. The Ambition is reviewed and refined at least annually, with discussions involving members of all Teams to align understanding and commitment.

The Goal translates the Ambition into actionable direction. It is a strategic objective that unites all Teams within an Arena around a shared outcome. Unlike individual performance targets that fragment effort, the Goal aligns everyone toward the same result. Multiple Arenas can share the same Goal, creating enterprise-wide alignment.

The Goal differs from individual targets in crucial ways:

- **Shared Direction.** All Teams within an Arena work toward the same Goal. There are no competing individual objectives that pull people in different directions.
- **Owner Accountability.** The Owner must ensure the Goal aligns with the Ambition and the Arena Backlog. This creates clear accountability for strategic alignment.

- **Flexible Duration.** A Goal spans one to nine Matches, allowing for both short-term focus and longer strategic initiatives.
- **Continuous Presence.** The Owner must ensure at least one Goal is always in focus. The enterprise never operates without direction.

The AME3 Response

Beside Goal and Ambition, AME3 addresses suboptimization through several integrated mechanisms. These reinforce each other: transparency reveals misalignment, flexibility enables correction, empirical validation proves the correction worked, and evolution focus ensures corrections serve the product's future.

- **Enterprise-Wide Transparency.** The Tournament provides regular inspection of the Enterprise Product by Accountable Representatives. This quarterly cycle ensures that local optimizations are evaluated against enterprise outcomes.
- **Structural Flexibility.** The Arena structure allows organizational boundaries to evolve with product architecture. System Leads work with Owners and Teams to experiment with structures that minimize dependencies.
- **Empirical Validation.** Through Empirical Control, AME3 ensures that optimization decisions are based on evidence, not assumptions. The Anticipate-Advance-Assess[20] loops provide regular feedback on whether local changes improve or harm the whole.
- **Evolution Alignment.** Evolution Focus ensures that optimization considers where products and services are on the evolution path. What works in Genesis differs from what works in Commodity.

Practical Implications

Overall Optimization requires shifting how enterprises think about improvement:

- **Slice Organizations Along Stable Interfaces.** When creating Teams or Arenas, choose boundaries that minimize ongoing dependencies. Stable interfaces reduce coordination overhead.

[20] see: Anticipate, Advance, Assess, 23

- **Slice as Late as Possible.** Every organizational boundary creates communication barriers. Delay creating new boundaries until the cost of not splitting exceeds the cost of coordination.
- **Consider System and Organization Together.** Technical architecture decisions are organizational decisions. Organizational decisions are technical architecture decisions. Make them together.
- **Optimize for Learning Speed.** In Complex Adaptive Systems[21], the ability to learn and adapt matters more than initial design. Shorter AAA-Loops provide faster feedback for enterprise-wide optimization.

The Second Doctrine

Empirical Control provides the mechanism for learning. Overall Optimization ensures that this learning benefits the entire enterprise, not just the unit running the experiment.

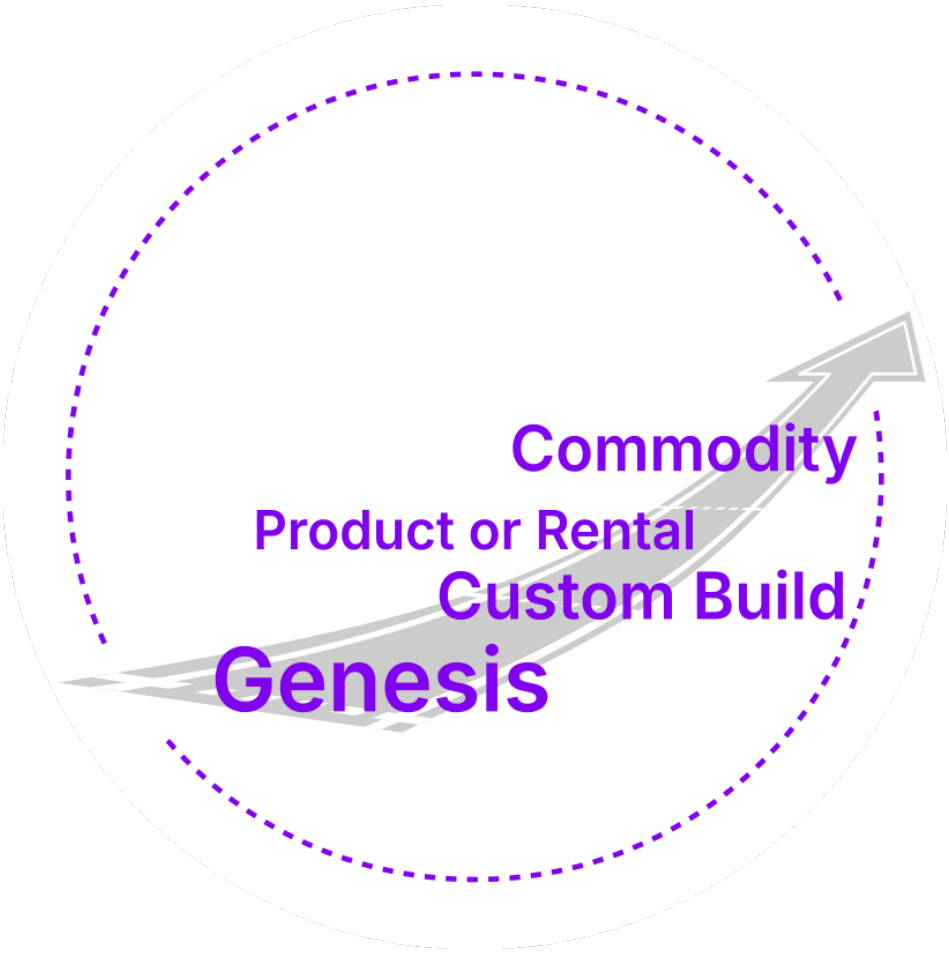
The goal is not to optimize the parts. The goal is to optimize the whole through the parts.

The challenge is continuous. Dependencies shift. Products evolve. Markets change. Overall Optimization is not a destination but a practice, a commitment to ensure that every improvement serves the enterprise, not just the unit making the change.

But knowing *whether* improvements help the whole is not enough. The enterprise must also understand *what* to improve and *where* to direct its energy. The third strategic doctrine, Evolution Focus, provides this context.

[21] see: Complex Adaptive System, 187

Evolution Focus



Navigating the Evolution of the Enterprise

In 2024, when I (Pit) started writing this book, I conceived the idea of an AI writing assistant to support my writing process in my preferred markdown editor. My idea was in its *Genesis* stage on the evolution scale of Simon Wardley. I soon discovered that others had similar ideas, and some had even developed plugins for my writing app. However, these plugins fell short of my specific needs. I decided to develop my own plugin, leveraging

a coding assistant based on a Large Language Model. This led to the birth of my *Custom Built* solution, which I called "AI Writing Assistant."

Reviewing these lines in early 2026, they already sound outdated. I have replaced my writing app with an armada of AI agents, scripts, and plugins. My agents now have self-learning mechanisms and improve themselves (see The End of Software as We Know It[22] for how this setup works in practice). Still, it remains a *Custom Built* solution, specific to my work as consultant, writer, and Co-CEO of my own company.

By the time you read this, your Apple device, Google Cloud services, or even your smartphone may already have integrated assistants far smarter and more capable than mine. That will present a challenge: maintaining my own solution to keep up with state-of-the-art functionality costs me time and prevents me from focusing on what I really want, writing about AME3.

It is even likely that the way we interact with software products has entirely changed by then. As described in The End of Software as We Know It, software interaction is evolving from tools we operate to agents that act on our behalf. The applications you use today may no longer exist as separate products. They may have been absorbed into AI agents that understand your situation and act before you ask.

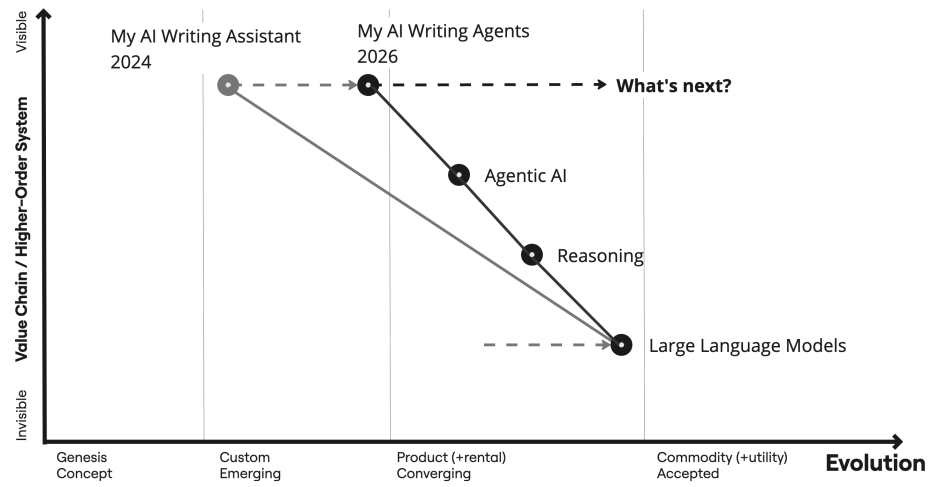


Figure 1.1.: Evolution of an AI Writing Assistant

[22]see: The End of Software as We Know It, 125

So what are my options now with my AI writing agents and assistants? I could work harder or code more efficiently, the most common improvement strategy. But focusing on an evolution strategy would bring me to a much higher level of innovation. I foresee two possible paths.

First, I can invest in changing my writing process and switch to an existing *Product* that I can buy or rent. I outsource parts of my writing process because the product adds more value than the money I spend. Alternatively, I have the option to offer my solution as a product, publishing it as open-source, enabling others to use and contribute to it.

Eventually, AI-based writing assistants will become a *Commodity*. We will integrate these tools into our workflow as seamlessly as we use water or electrical energy. Until then, many AI products and services will emerge and disappear from the market. As customers, we will experiment with them and eventually establish a standard for their integration into our daily lives, the same way we once did with pen and paper. **The decisions for your products and services today are creating the challenges of tomorrow.**

Evolution at Enterprise Scale

My AI writing assistant is a small example. Let me share a larger one.

In 2004, Andy hired me at Web.de, at the time one of the largest internet portals in Germany, and the beginning of a collaboration that eventually led to this book. Web.de served roughly 80% of all German internet users. Millions had an account and used the mail service daily. The customer base was growing exponentially, driven by broadband adoption, and new technologies were rising even faster.

The computing and network infrastructure that my colleagues maintained one floor below, where I was developing the services, filled several machine rooms with thousands of servers. Dedicated teams managed servers, storage, and network equipment around the clock. This infrastructure was *Custom Built*: designed, assembled, and operated specifically for Web.de's services. It was expensive, it required specialized knowledge, and it was the foundation everything else depended on.

Twenty years later, that complete infrastructure fits into a virtual machine I can rent within seconds from any data center in the world. What once

required dedicated teams and physical hardware has become a *Commodity*, available on demand at marginal cost. Simon Wardley documented this pattern across the entire history of computing infrastructure.

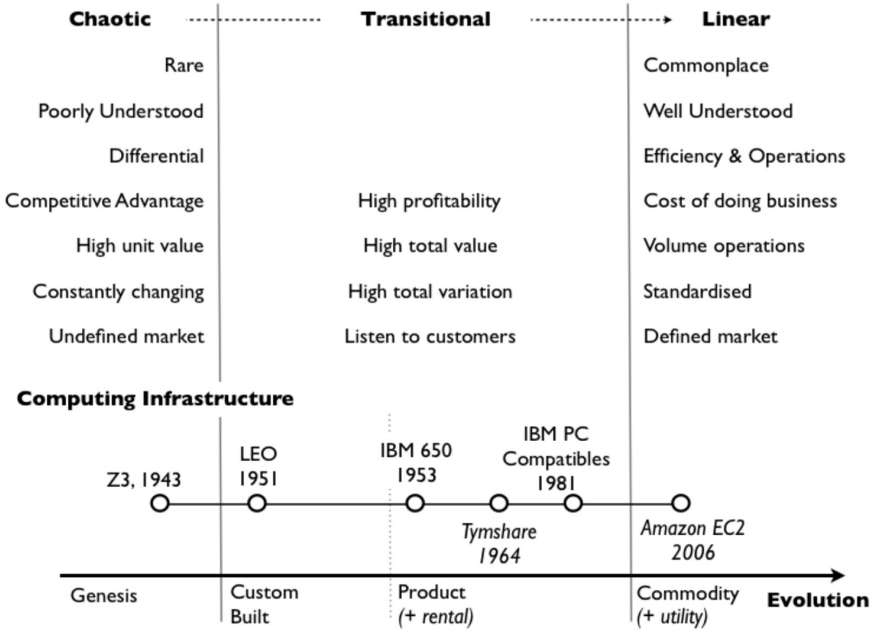


Figure 1.2.: Evolution of Computing Infrastructure (Simon Wardley, "Future 'is' predictable")

This is the evolution path that all products and services follow. Wardley describes four stages: *Genesis* (novel, uncertain, requires exploration), *Custom Built* (understood enough to build for specific needs), *Product* (mature enough to offer to a broader market), and *Commodity* (standardized, available as utility). My AI writing assistant and Web.de's computing infrastructure are examples of the same force at work, at very different scales. Every Enterprise navigates this path with every product and service it offers.

A Universal Principle

Evolution is not limited to technology or software. As explored in AI and the Principles of Evolution[23], it is a universal principle: every major technology, from the printing press to electricity to the internet, led to greater complexity, not simplicity. Our products and services follow the same pattern. Their components and parts will continue to advance as long as our society invests energy in our markets.

Evolution is not a constant flow. Tushman and O'Reilly showed that organizations evolve through periods of incremental change punctuated by discontinuous, revolutionary change (Ambidextrous Organizations[24]). The development of Artificial Neural Networks illustrates this pattern vividly. There was a long period, known as the AI winter, where innovation was stagnant. Then, with the availability of vast amounts of data, computing power, and substantial investment capital, progress accelerated dramatically. **Enterprises must be prepared for both: long stretches of steady evolution and sudden disruptions that reshape entire industries.**

The evolution of any individual component like a Large Language Model[25] can take decades. But Wardley identifies a Climatic Pattern[26] that changes the picture: the evolution of communication increases the speed of evolution overall. Every advance in how we share knowledge, from the printing press to the telephone to the internet, compresses the cycle for everything that builds on top of it. AI is the latest and most powerful step in this sequence.

What makes the current moment feel different is that multiple evolutionary waves now overlap. AI, robotics, biotechnology, and energy transition each follow their own timeline, but together they create an environment where disruptive change arrives from many directions at once. In the early part of the last century, the span of disruptive innovation often exceeded an engineer's career. Now, as an engineer over 50 years of age, I find it challenging to count the groundbreaking innovations I have had the privilege

[23] see: AI and the Principles of Evolution, 4
[24] see: Ambidextrous Organizations: Managing Evolutionary and Revolutionary Change, 223
[25] see: Large Language Model, 226
[26] see: Climatic Patterns by Simon Wardley, 190

to witness. **We cannot stop evolution. It is in human nature. But we can lead it.**

Agility and Efficiency Are Not the Goal

Evolution does not stop at the product. Every shift along the evolution path changes how we need to work. A Team exploring a novel idea in the Genesis stage needs freedom to experiment, fast feedback, and the ability to pivot. A Team operating a Commodity service needs reliable processes, consistent quality, and operational stability. The product evolves, and the work system must evolve with it.

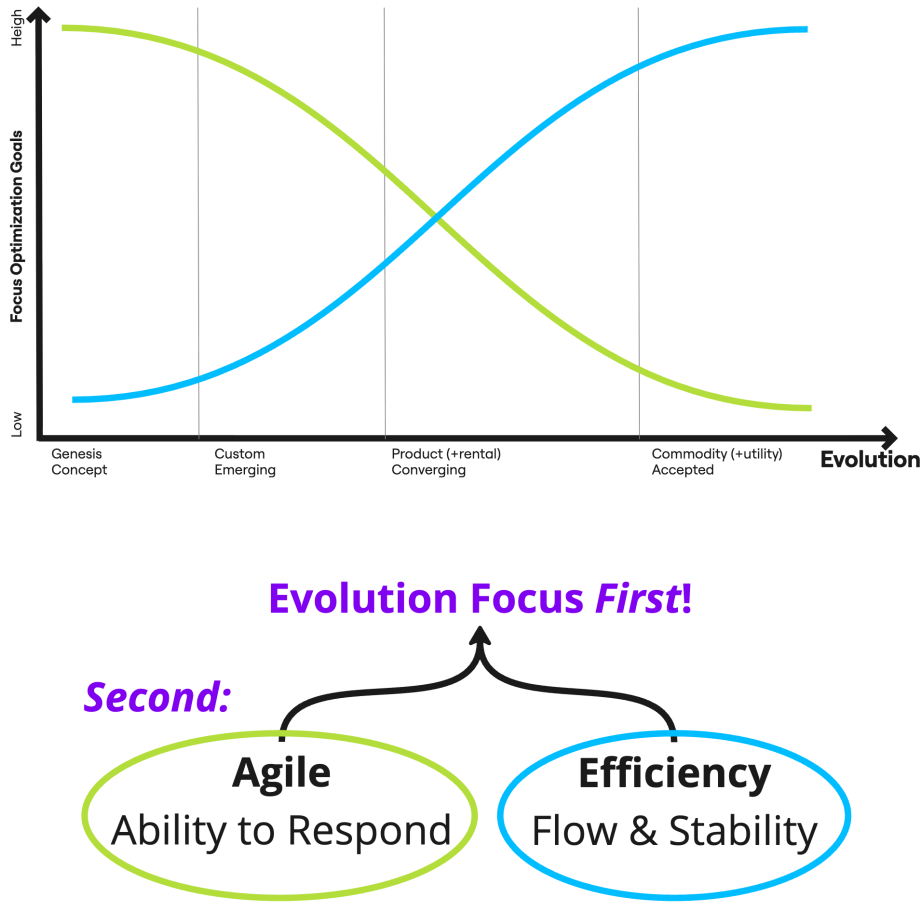


Figure 1.3.: Focus on evolution first. Agile and efficiency come second.

This is why many organizations get stuck when they pursue agility or efficiency as their primary goal. These are valuable, but they are lower-order goals. As the image shows, agility (the ability to respond) matters most in the early evolutionary stages: Genesis and Custom Built. Efficiency (flow and stability) matters most in later stages: Product and Commodity. A water utility delivering commodity services should optimize for consistent quality and reliable delivery, not for agility. A startup exploring a novel idea should optimize for learning speed, not for efficiency. However, most organizations operate somewhere in the middle, with products and services

at different evolutionary stages simultaneously. They must balance both agility and efficiency, applying each where it fits.

Different parts of the organization sit at different evolutionary stages. They require different optimization strategies, and therefore different designs for their work systems. Only an evolution-based approach reveals which balance is right and where to focus. Agility and efficiency are consequences of making the right evolutionary choices. They are tools that serve different phases, not ends in themselves. **The question is not "how efficient are we?" or "how agile are we?" but "how well do we navigate evolution compared to our competitors?"**

Reorganization as Standard Practice

Whenever we advance our products and services along the evolution path, reorganization becomes necessary. New products in the Genesis stage need small, cross-functional Teams with the freedom to explore. As products mature into Custom Built solutions, Teams grow and specialize around stable boundaries. When products become Commodities, the organization must shift toward operational efficiency, or consider outsourcing entirely. Each transition changes team structures, leadership responsibilities, and the methods that work best.

While this may seem dramatic, it is standard practice within an AME3 organization. Maintaining flexibility is embedded in the Rules and the Agile and Lean methods we employ. Understanding the evolutionary phase of the Arena Product guides Teams and their System Leads in selecting the most appropriate methods and frameworks.

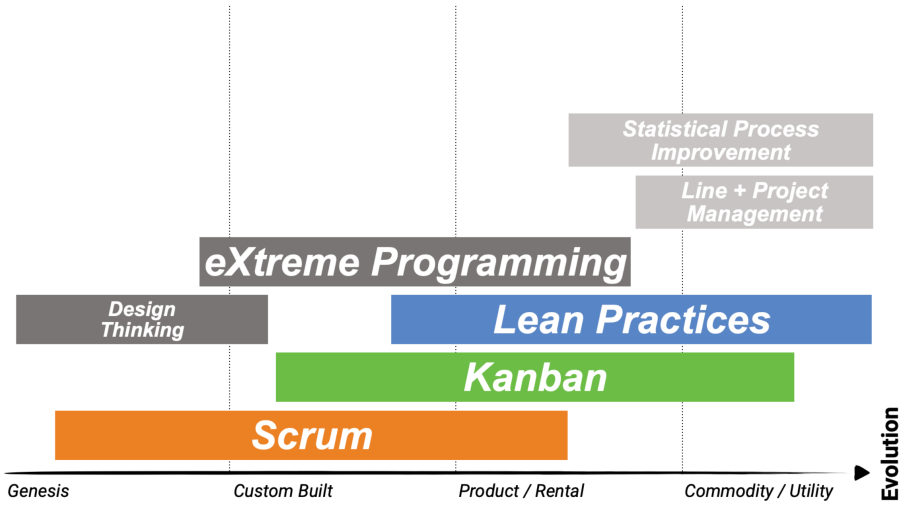


Figure 1.4.: Best use of practices and frameworks

Evolution Focus is a strategic doctrine in AME3, exercised through the co-creation process with the experts in the Teams, the Owner and Enterprise Owner, facilitated by the System Leads. It is not a one-time assessment but a continuous practice, revisited in every Tournament as Accountable Representatives assess where the Enterprise Product sits on the evolution path and where to direct energy next.

The Third Doctrine

Evolution Focus completes the strategic foundation of AME3. Where Empirical Control establishes the mechanism for learning and Overall Optimization ensures that learning benefits the whole, Evolution Focus answers the deeper question: what should we be learning about, and where should we direct our energy?

Without Evolution Focus, enterprises risk optimizing in the wrong direction. They may become highly efficient at producing something the market no

longer values. They may become highly agile at responding to change without understanding which changes matter most.

Evolution Focus ensures that Teams, Owners, and System Leads make informed decisions about innovation, outsourcing, and organizational design based on where their products and services sit on the evolution path.

Together, the three doctrines create a self-improving system:

- **Empirical Control** answers: How do we know if we are improving?
- **Overall Optimization** answers: Who benefits from the improvement?
- **Evolution Focus** answers: What should we be improving?

An enterprise practicing all three doctrines generates data through local experiments, ensures alignment across the whole, and directs energy where evolution demands it. The next section, Leadership[27], explores how AME3 structures the leadership functions that bring these doctrines to life.

[27] see: Leadership, 47

2. Leadership

Every enterprise is a leadership system, whether designed or not. People lead each other in making decisions every day, at every level. The question is not whether leadership exists, but whether it serves your goals.

Most enterprises struggle with leadership not because they lack leaders, but because their leadership lacks structure. Decisions stall in committees. Local targets conflict with shared outcomes. Growth creates confusion instead of clarity.

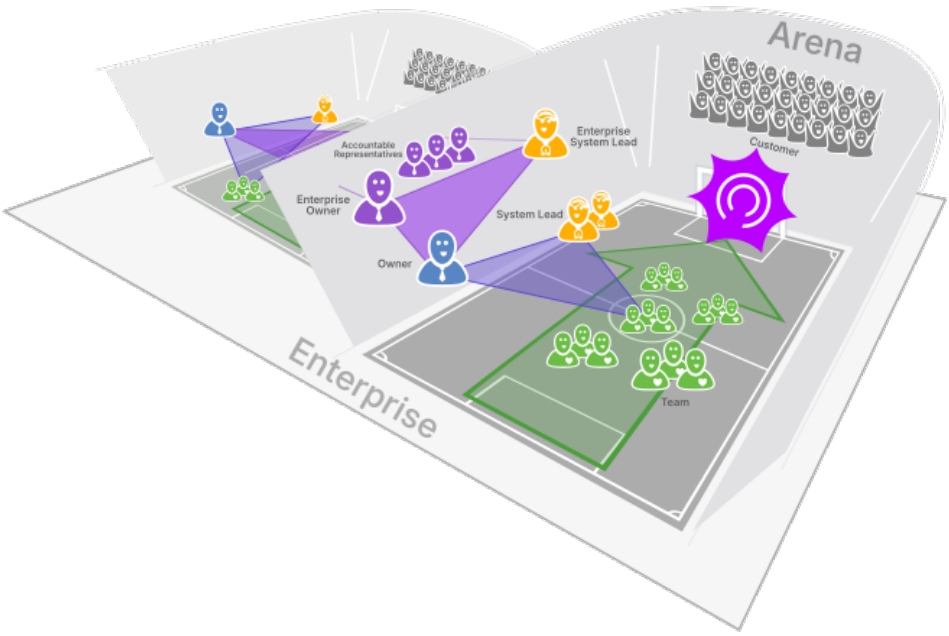
These problems do not resolve by adding more managers or processes. They resolve by designing a system that channels leadership where it matters.

AME3 provides such a system, built on a triangle of forces: Owners drive the success of products and services, System Leads build effective work systems, and Teams deliver customer satisfaction. Building on the strategic doctrines[1] and together with Rules, this balanced structure creates stability without rigidity. It is self-regulating yet highly adaptable to changing markets.

The Leadership System[2] explains how these three functions work together as checks and balances. The following chapters explore each function through the stories of two companies: a start-up and a medium-sized enterprise.

[1] see: Strategy, 19
[2] see: The Leadership System, 48

The Leadership System



The Essence of Leadership

You might be familiar with this scenario: A young girl, possibly around five years old, stands in line with her father at a supermarket checkout. She's captivated by one of the enticing offers placed at her eye level. With wide, pleading eyes, she turns to her father and asks, "Can I have that pleeeeaaase?".

Yes, I was the father in that story, and my response was: "OK, but let's keep it a secret from your mother."

The scene took place in 2006. At the time, I was deeply reflecting on the meaning of Agile Leadership. A term gaining traction in the business world at the time, as many other "old" terms did by adding Agile before them. In the parking lot, it struck me: what had my daughter actually done? She used her voice and body language to lead me to a decision that served her goal.

So, she was actually *leading* me. I felt ashamed because, as a father, I expect myself to lead my children in making wise decisions and selecting healthy options.

But, thanks to my daughter, I found for myself a definition for the term leadership at this moment:

> *Leadership is essentially the way to lead others in making decisions to achieve a greater goal.*

There are many ways to do this. My daughter's way can perhaps be described as the *big-eye* method. Command and control or servant leadership are others. All have their place and belong to the collection of leadership practices.

The story also told me, that leading people happens all the time, every day, by everyone. By writing this article, I am leading you as a reader to insights that will help you make decisions for yourself and your business.

In *The Team*, I stated: You cannot not lead. If a team member backs down when discussing a decision (perhaps because they shy away from conflict), they lead to a decision. Even if they do not support this decision.

Leadership is ever-present and inherent; it doesn't need to be implemented. *Leadership* ends with the suffix *-ship,* like in *mastership,* and denotes the way things are done. Thus, discussing leadership is essentially discussing our methods of leading.

What is Agile Leadership, and why Do We Need It?

Returning to the concept of *Agile Leadership*, it highlights the necessity for a distinct understanding of *leadership* to establish and maintain an agile organization. Let's explore this further.

In 2001, a small group of people defined values and principles for software development and recorded them in the Manifesto for Agile Software Development[3]. This set of values and principles is generally credited with popularizing *Agile* in the working world. For the authors, the term *Agile* mainly expressed why they would work according to these values and principles.

[3]see: Manifesto for Agile Software Development, 203

Software development has had to react to the ever-faster-changing market and technology development. Other domains today, such as software development in 2001, are facing a similar challenge. Digitalization, Industry 4.0, Generative AI, and new work are just some of the buzzwords. Agile is, therefore, an optimization goal for the organization. In *The Team*, I provided a very brief definition for agile: "Agile is the ability to respond to change."

So, to summarize, we can say:

> *Agile leadership is the way to lead others in making decisions facing a rapidly changing environment, adhering to the values and principles of the Agile Manifesto.*

But how do you get this kind of leadership up and running in your enterprise? The answer is, you need a system embedding this way of leadership.

What is a Leadership System?

Simply put, if we accept that You cannot not lead[4], all social groups operate as leadership systems. Recognizing this, the term *leadership system* implies that to achieve a specific leadership style, we must create the appropriate system that allows that leadership to thrive.

So, to be an agile leader, you must first develop such a system and live it yourself. AME3 can help you.

How Does the Leadership System Work within AME3?

AME3 sees itself as a framework. As such, it defines a skeleton with three functions, their fundamental responsibilities, and the constraints on which the system can work. It intentionally does not offer a complete operational model. This approach allows the system to evolve and adapt over time.

The rules are based on decades of observation of Agile organizations by a large community. One member of this community, Nigel Baker, would declare them as core practice on his Nigel Scale[5]. Something you must do

[4] see: The Team, 63
[5] see: Nigel Scale, 203

no matter what. Just as a surgeon always disinfects their hands before entering the operating room without questioning it.

Viewed together, the functions form a leadership system that resembles a triangle of forces. This system is stable enough not to swing to the extreme if one force gets out of hand. This addresses the enterprise's need for control.

On the other hand, designing a system based on only three forces maximizes agility by minimizing delays caused by waiting for others during decision-making. This approach addresses the need for high flexibility in continuously changing environments, such as dynamic markets.

In other words, AME3 forms a system of checks and balances. It is stable and self-regulating, yet with maximum adaptability. Such triangles of forces have proven their worth in social systems many times, e.g., in governing a state: legislation (legislature), executive power (executive), and jurisdiction (judiciary). Or in running a football club: team, coach, and club president.

The Leadership Functions in a Nutshell

The Leadership System of AME3 is fundamentally composed of three key functions: Owners, System Leads, and Teams.

System Leads lead to an **effective work system.** They serve the enterprise by:

- Developing competencies of Teams and people.
- Facilitating decision-making.
- Sustaining a continuous cycle of Anticipate, Advance and Assess.

Owners lead to the **success of the product and services.** They serve the enterprise by:

- Balancing opportunities and risks.
- Focusing the organization to increase effectiveness.
- Ensuring decisions are made.

Teams lead to **customer satisfaction.** They serve the enterprise by:

- Managing and executing the work.

- Creating value and ensuring quality.
- Identifying opportunities for improvement in products, services, and work systems.

Conclusion

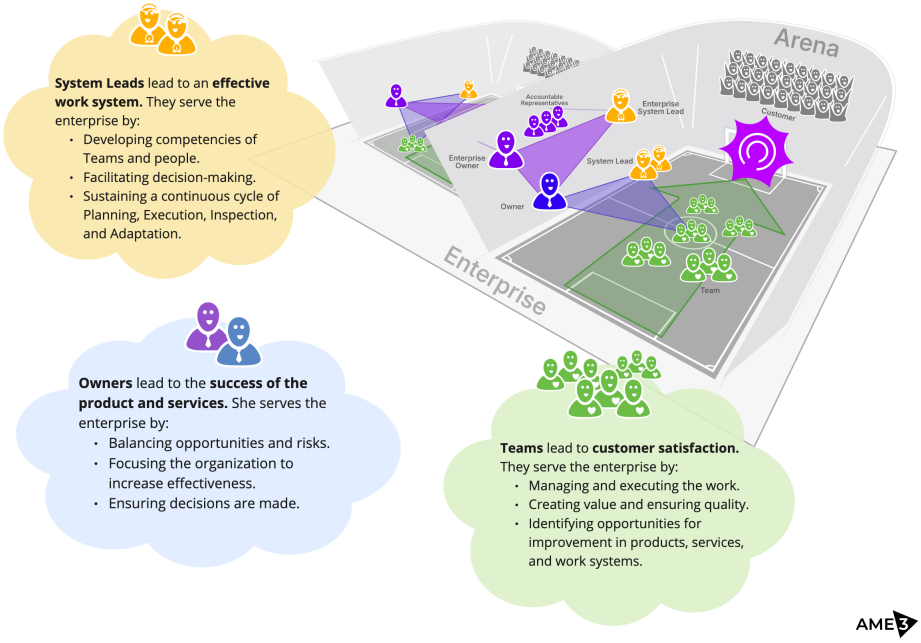
Leadership is the process of leading others in decision-making to achieve a greater goal. Agile Leadership is necessary for maintaining an adaptable organization. AME3 helps establish and maintain an Agile Leadership System that is stable and self-regulating yet with maximum adaptability. It consists essentially of three functions: Owners, System Leads, and Teams.

The Owner

Leading to Success of Product and Services

AME3 incorporates a leadership system, with the Owner being one of its three main forces. To illustrate the leadership role of the Owner, let's examine two different companies.

The first is a nimble start-up with 20 employees, while the second, a well-established medium-sized Enterprise, employs 1200 people. Both use AME3 to navigate their unique challenges towards growth.



Example: Start-up

Ownership is leadership with power.

Our example start-up was founded a few years ago with a small Team. The founder, who is also the main shareholder, is integral to a Team of now 20 employees and continues to play a pivotal role in product development. Recently, he secured additional investment capital aimed at accelerating growth.

As the leading expert in the technical domain, the founder now faces the critical challenge of empowering his employees to reduce their reliance on his expertise. This transition is essential for him to shift his focus entirely to leading the business, thereby ensuring the company's sustained growth.

The new product has the chance to dominate the global market, with immense opportunities and minimal competition. As the founder and co-owner, he is responsible for defining the Ambition and setting intermediate Goals for product development.

Within the context of AME3, our founder is the Owner, and his employees form the sole Arena of his Enterprise.

The work environment is highly complex, with constant initiation of new projects. This creative chaos can often lead to frustration without clear guidelines. The employees are demanding concrete direction to maintain focus. Our Owner must collaborate with the Teams to identify the most lucrative opportunities in the market. The Arena Product is teetering between the concept stage and Custom Built in its evolution.

The Teams are seeking answers to critical questions. For instance, how can they test new functionalities using straightforward experiments without significant investment in development?

Furthermore, architectural decisions made now will significantly influence the company's growth, presenting both opportunities and risks that the Owner must manage meticulously.

Given the complexity of their work, the employees adopted Scrum from the outset and organized themselves into Teams following the Large Scale Scrum (LeSS) framework. The Owner engaged an external System Lead to facilitate this process. Since Scrum and LeSS are aligned on principles with AME3, this transition was seamless.

The group of co-owners and investors make up the Accountable Representatives. The business case presented by the Owner serves as the Ambition for the Arena and its Arena Product.

Example: Medium-sized Enterprise

In our medium-sized enterprise, the original founders have transitioned to the supervisory board, entrusting decision-making authority to a newly appointed CEO[6]. Her primary objectives are to enhance profitability and drive innovation to stay competitive in a rapidly changing market.

To achieve these ambitious goals, the CEO plans to enhance the organization's overall agility using the AME3 framework. Within AME3, the supervisory and management boards together constitute the Accountable Representatives.

[6]see: Chief Executive Officer (CEO), 186

The Arena Product is the result of all services, with the goal of maintaining and increasing value for the customer.

Assuming that the Enterprise has just one Arena as in the example of the start-up, and we view the enterprise's results as a single Enterprise Product, we have to accept that the CEO is solely responsible for all investment and the overall success of it. We can recognize the CEO as the sole Owner or Enterprise Owner in terms of AME3.

Recognizing the impossibility of leading 1,200 employees alone, it is clear that the CEO plans to divide the product into sub-products, reorganize the company into distinct Arenas, and delegate her ownership responsibility. This aims to create a lean hierarchy, empowering each Arena to independently optimize its Arena Product.

However, this is all theoretical, and the organization is far from this ideal.

The current organizational structure emphasizes technical and functional specialization, where career progression and employee recognition are tightly woven into the hierarchical system. This has led to a leadership model focused on maximizing work efficiency through specialized roles and cost reduction. Consequently, the multi-layered decision-making process is sluggish, impeding the organization's responsiveness to rapid market changes.

Moreover, the existing line management has resulted in conflicting objectives. For instance, the head of product management prioritizes sales growth by demanding new features, while production and operations focus on reliability. This creates tension between the departments. This model inherently resists innovation, clinging to the status quo.

The previous CEO introduced project management to boost innovation across functional barriers. However, this created additional management roles with conflicting goals, worsening the problems.

The diminishing number of value-creation experts, overwhelmed by an average of 4–5 projects each, are increasingly frustrated, leading to burnout and high turnover. This problem is starkly reflected in a portfolio of over 120 active projects, most showing little or no progress.

Recognizing the urgency, the new Owner is striving to significantly trim the number of projects. This task is challenging, as many projects have broad scopes and encompass both high value and high risk.

Collaborating with the Accountable Representatives, including experienced managers and Team experts, the CEO initiates a strategic refinement process using Wardley Mapping. This method helps identify necessary improvements in Products and services, alongside required changes in the organization. Critical objectives from the project portfolio are integrated into this strategy too and added as Goals to the Enterprise Backlog.

This process also identified potential new Arenas and their potential Owners. Some of these Arenas already exist as organizational units, while others need to be created by reorganization. All these changes are innovation Goals too and are added to the new Enterprise Backlog.

To outline their strategic plan, the CEO ordered the Goals in the Enterprise Backlog and, together with the workgroup, pulled a very limited number of Goals into focus.

An external expert in organizational design facilitates this entire process, introducing and implementing the necessary methodologies, including a Kanban[7] system, to manage the Enterprise Backlog and Goals. Later, they will take over the responsibility of the Enterprise System Lead.

Rules

AME3 establishes rules that foster the development and flourishing of the leadership system. Here are the rules connected with the Owner and Enterprise Owner:

Owner

The Owner is the leadership function responsible for the success of the Arena Product in alignment with the enterprise Ambition. They hold authority over Arena Backlog ordering and capacity decisions. The Owner collaborates with System Leads while maintaining distinct accountability.

- The Owner leads to the success of the Arena Product, in line with the Ambition.
- The Owner determines how the Arena Backlog is ordered.

[7] see: Kanban, 199

- Only one person serves as the Owner, who cannot simultaneously be a System Lead.
- The Owner sets the maximum number of employees in the Arena based on operational constraints.
- The Owner must consider the System Lead's recommendations.
- The Owner is an Accountable Representative.

Enterprise Owner

The Enterprise Owner is the executive leadership function responsible for the success of the Enterprise Product. They have the authority to initiate or stop Arenas, change Ambitions, and order the Enterprise Backlog while considering Enterprise System Lead recommendations.

- The Enterprise Owner leads to the success of the Enterprise Product.
- The Enterprise Owner can initiate or stop an Arena and/or its Product.
- The Enterprise Owner can change the Ambition of an Arena.
- The Enterprise Owner is an Accountable Representative.
- The Enterprise Owner cannot be a System Lead or Enterprise System Lead.
- The Enterprise Owner must consider the Enterprise System Lead's recommendations.
- The Enterprise Owner determines how the Enterprise Backlog is ordered.

Conclusion

In both scenarios, the Owners lead to the success of products and services. They serve the Enterprise by:

1. Balancing opportunities and risks.
2. Focusing the organization.
3. Ensuring decisions are made.

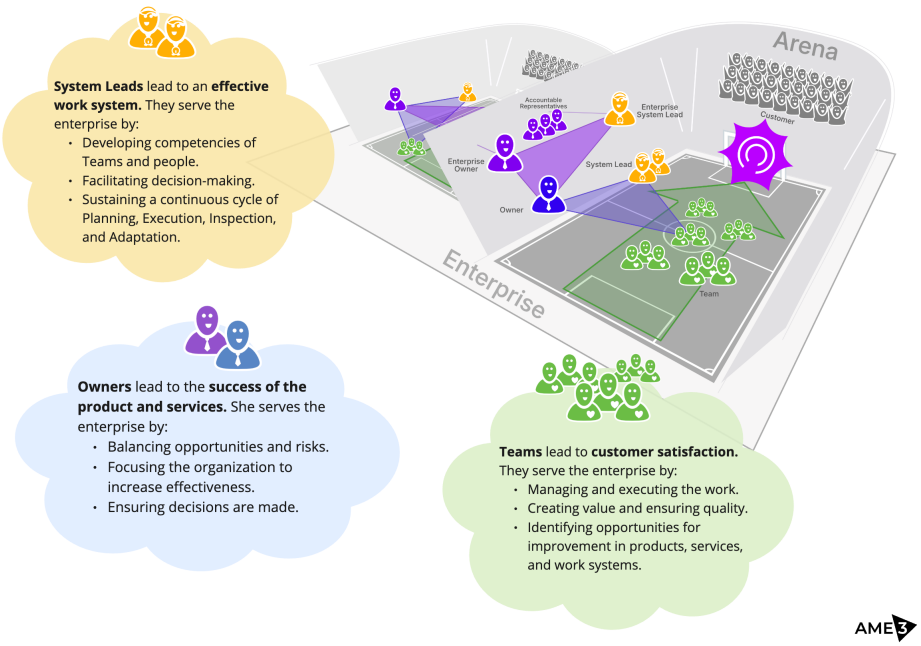
AME3 empowers them to make informed decisions and delegate effectively, fostering an adaptable culture and continuous improvement.

The System Lead

Enhancing Effectiveness of the Work System

In the dynamic landscape of modern business, effective leadership can make or break an organization. AME3, a progressive leadership framework, empowers businesses to navigate complexity and drive innovation through three fundamental functions: Owner, System Lead, and Team.

Building on the previous chapter surrounding ownership responsibility[8], we now shift focus to the catalyst for continuous improvement: the System Lead.



The System Lead embedded in the Leadership System of AME3.

Through the lens of two examples — a burgeoning Start-up and a medium-sized Enterprise — we will explore how System Leads within AME3 enhance effectiveness and adaptability.

[8] see: The Owner, 52

Example: Start-up

> Culture follows structure – Larman's Laws[9]

From the beginning, the start-up implemented AME3 as its management system, thanks to one of its first employees. She convinced the founder to adopt an Agile[10] and flexible corporate structure to facilitate the early growth of the enterprise. Additionally, she maneuvered the founder into the function of the Owner or, while she took on the role of System Lead.

While AME3, unlike Scrum, is not exclusively designed for the complex domain, it recommends Scrum for the given situation. Scrum defines specific roles for a team working in complex environments. It's perfect as the leading framework for a start-up in its early stages. The roles of Owner, System Lead, and Team in AME3 align on the principle level with the Scrum roles of Product Owner, ScrumMaster, and Developer.

So, as the organization grew beyond a single Scrum Team, establishing essential functions such as finance and HR alongside the development teams became straightforward. Within AME3 she could easily integrate other frameworks and methods. In her case, Large Scale Scrum for the development and Kanban to organize the basic corporate functions.

The company has since grown to 50 employees. Growth also means the development of the teams and people skills. It is the System Lead's job to constantly develop these further. Teams had to be newly formed and built up with new employees. New practices had to be learned, and old ways of doing things had to be unlearned.

The System Lead also had to act if an employee couldn't be integrated into the Teams permanently, sometimes leading to parting ways with them. She handled these actions personally. For other matters, she facilitated the process. Consequently, decisions on new hires were made collectively by team-representatives, the Owner, and the System Lead.

As the number of Teams increased, so did the number of System Leads. At the time in question, four more System Leads were grouped around the first System Lead. There are various focal points to which the System Leads are dedicated. For example, facilitating the recruiting process, introducing and adapting engineering methods, continuous coaching on product strategy.

[9] see: Larman's Laws, 200
[10] see: Agile, 185

> The product architecture is a reflection of the organizational structure that develops the product. – Conway's law

One of the most enduring challenges is the strong dependency and the necessary communication between the Teams. What was still somehow feasible with 30 employees is now starting to become a real problem. More and more isolated solutions are being developed by the Teams which are not integrated into the common Arena Product, and therefore are not usable by the customers. So, although the organization has grown in employee numbers, the overall performance has not improved.

Teams often opt for isolated solutions to avoid waiting on other Teams for results or decisions, a problem that intensifies with each new hire. The only solution for future growth is to design the Arena Product architecture so that Teams have minimal overlapping activities. Furthermore, it's crucial to enhance the Teams' self-organization and engineering skills.

The System Leads must therefore work very closely with the Owner and the Teams to repeatedly conduct experiments with them to promote new insights for the Arena Product architecture, organizational structure and practices.

So, the System Leads together as a unit have taken on the responsibility of optimizing the entire work system, which we call Arena in AME3.

Example: Medium-Sized Enterprise

In the first part of the series, we observed how the Enterprise Owner developed and prioritized the Enterprise Backlog, focusing the organization on a few key Goals. Initial Arenas and Products utilizing the AME3 structure were also identified. This strategic plan was created with input from Accountable Representatives and experts, guided by an experienced external consultant.

At the end of this process, the Enterprise Owner offered the external consultant a continuous role as System Lead to facilitate the process of inspecting and adapting the enterprise strategy on a permanent position, making him the Enterprise System Lead. His initial task was to align the existing board and executive meetings.

The refined structure now integrates a strategy workshop with all Accountable Representatives every three months, coupled with a weekly alignment

meeting that includes the Enterprise Owner, Owners, and the executive management. In AME3, this enterprise wide cycle of inspection and adaptation is termed as a Tournament.

During this initial process, 2 potential Arenas and their corresponding Products were identified:

1. The division dedicated to the development and Production of a globally successful laser welding tool family.
2. A novel machine tool that potentially leverages a new approach to 3D printing, synergized with cutting-edge AI-based control software.

From the enterprise's perspective, the two highlighted Products bore striking resemblances, as they both held immense strategic value.

In addition to these considerations, the following factors significantly influenced the selection of the **first Arena Product**:

1. The development and production of the laser welding devices were able to operate relatively independently of other organizational units without significant reorganization.
2. The dependencies on suppliers were already managed by the unit, so changes could be managed directly.
3. The development already used Scrum, and the production had experimented with Lean Management practices. However, a comprehensive approach was yet to be established, which is precisely what an Arena in AME3 is offering.

In collaboration with the ScrumMasters and people leads, the Enterprise System Lead formulated an improvement plan for the Arena. This plan encompassed the establishment of Matches for the entire Arena, the creation of an Arena Backlog, and aligning the Arena with the enterprise's Tournament. The Owner of the Arena, previously the head of production and now also accountable for development, added these measures to the Arena and set the final priorities. At a later point, ScrumMasters and people leads reorganized. Some transitioned into System Lead function, while others filled the gaps as highly needed experts within the Teams.

The **2nd Arena Product**, the 3D printer machine tool, hardly differed from the start-up example. Instead of a CEO of the start-up, the Owner of

the Arena was a former product manager with a strong engineering background. The System Lead's role was taken on by an experienced ScrumMaster, who was a fresh hire by the Enterprise System Lead and the Owner.

Within the rhythm of the Tournament, the Enterprise System Lead and System Leads of the two Arenas continuously inspect and adapt their organizational development work. This process is always aligned with the Arena Product goals set by the Owners and Enterprise Owner. It also includes Team and people development activities, such as exchanging experiences between the two Arenas.

This leads us to the pivotal leadership function within AME3. In Part 3, we will explore how Teams drive towards achieving customer satisfaction.

Rules

AME3 establishes rules that foster the development and flourishing of the leadership system. Here are the rules connected with the System Lead and Enterprise System Lead:

System Lead

The System Lead ensures an effective work system within the Arena and guarantees compliance with AME3 rules. They establish and enforce additional frameworks and methods as needed. Multiple System Leads can serve an Arena, collectively accountable for all Teams.

- A System Lead leads to an effective work system in the Arena.
- A System Lead ensures compliance with the AME3 rules.
- An Arena can have multiple System Leads.
- The System Leads are entitled to establish, discard, and demand compliance with rules beyond the AME3 Framework. This includes additional frameworks and methods.
- A System Lead cannot simultaneously serve as the Owner and is appointed by the Owner.
- A System Lead may specialize in certain Teams, yet collectively all System Leads remain accountable for all Teams.

Enterprise System Lead

The Enterprise System Lead is the System Lead for the enterprise-level work system in AME3. This leadership function ensures effective structures and processes across all Arenas while maintaining system coherence. They can also serve as System Lead for individual Arenas.

- The Enterprise System Lead is the System Lead for the work system on Enterprise level.
- They can also be System Lead for an Arena.

Conclusion

The **System Leads** lead to an **effective work system.** They serve the Enterprise by:

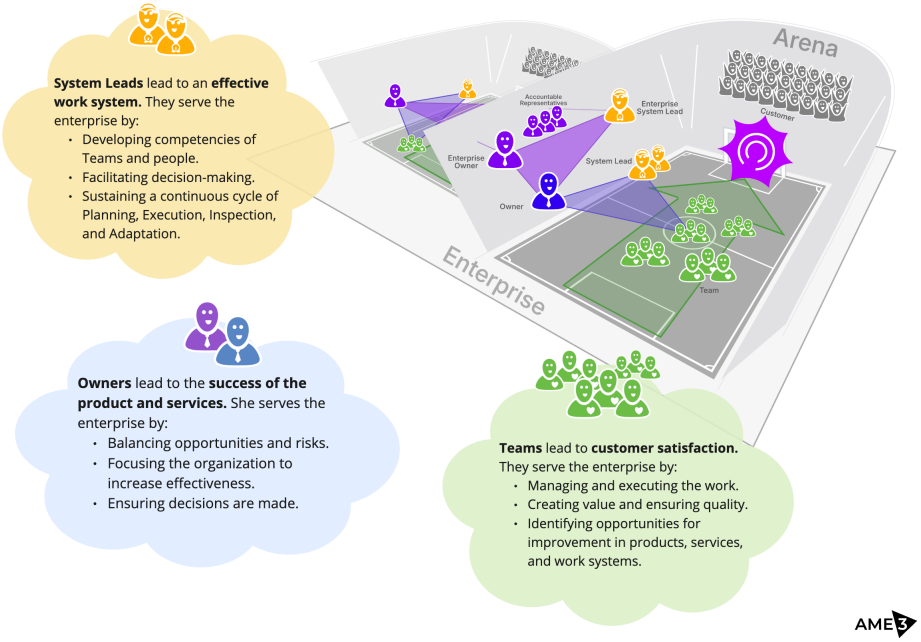
- Developing competencies of Teams and people.
- Facilitating decision-making.
- Sustaining a continuous cycle of Planning, Execution, Inspection, and Adaptation.

Uncover the ongoing journey of the two enterprises in our forthcoming article, which will spotlight the Leadership role of the Teams.

The Team

Leading to Customer Satisfaction

In the previous two chapters, we highlighted the leadership of System Leads and Owners through the narratives of two companies. This part will concentrate on the often underestimated leadership function of the Team, further enriching these stories.



Example: Start-up

> What motivates people: Autonomy, Mastery, and Purpose – Drive, by Dan Pink[11]

In the early years of the start-up, hardly anyone thought about leadership or management. Everything seemed to happen naturally and felt quite organic.

New ideas were pursued with vigor when they ignited a spark of excitement within the Team. Given that the purpose of any activity serves as a significant catalyst for motivation, every Team member was naturally driven to cater to customer needs, all in the pursuit of positive feedback.

Surely, in many cases, it was more about experimenting. Whether there was indeed a customer need was just a hypothesis. Likewise, new technical solutions were passionately developed, tested, and often discarded. Experts were consulted. If the proposed solutions were suitable, these experts often joined the Team, as enthusiasm is known to be contagious.

[11] see: Drive, by Dan Pink, 190

> You cannot not lead. – Inspired by Paul Watzlawick[12]

Parts 1 and 2[13] showed that this naturalness was in danger of being lost after the period of early growth. AME3 in conjunction with the application of frameworks like Scrum and later the LeSS framework helped the employees preserve the naturalness of the early years.

The organization discovered that work flows more efficiently when Teams are empowered to make timely decisions independently. This autonomy extended to their commitment to creating value for customers.

Thus, it remains entirely self-evident that all the work is self-managed by the team members. Especially when this means close coordination with other Teams, customers, and suppliers.

Therefore, the System Lead ensured that every employee received leadership and communication training. This significantly contributed to the Teams' performance improvement.

> Not the Owner leads the Team through decisions, but the Team leads the Owner to the decisions

As the company expanded, the primary focus of creating solutions that enhance customer satisfaction remained with the Teams. Yet, customer satisfaction isn't the sole interest of a business. Our start-up also had to make tough decisions that weren't always in favor of all potential customers to maintain profitability and focus. Naturally, these decisions were made by the Owner. However, the data and intelligence needed for these decisions, and even the identification of which decisions needed to be made, came from the experts within the Teams.

The Owner and Teams engaged in a continuous feedback loop of Matches (Sprints) and Tournaments. This process, facilitated by System Leads, helped them develop a nuanced understanding on how to delegate. They learned who needs to make decisions, how these decisions should be made, and when to involve or inform others.

[12] see: Paul Watzlawick, 205
[13] see: The System Lead, 58

Example: Medium-Sized Enterprise

The Team's situation within the medium-sized Enterprise varies significantly from one Arena to another, and even within the Arenas themselves.

The Arena, focused on the 3D printer machine tool as a Arena Product, shares many similarities with our start-up example, despite the product itself being of a completely different nature. When this Arena was created, new Teams were formed with employees from other parts of the enterprise and by hiring new experts. Old rules and structures ceased to exist from one day to another. With a fresh start in mind, employees readily embraced and championed this new approach to work. This process also developed a strong sense of customer satisfaction by the teams.

The Arena of the laser welding tool family, already familiar with Lean and Agile methods, found the transition to AME3 initially subtle. However, the new overarching structure began to address deeply rooted problems tied to team dependencies. They now had a platform to address these issues during the Anticipate, Advance and Assess phases of Matches and Tournaments. Yet, team reorganization, and the learning and unlearning of practices, remain an integral part of this routine.

Since the establishment of the first two Arenas, three more have been thoughtfully created in line with the Enterprise's strategy for evolution. A significant challenge for these Arenas is that employees are organized into specialized groups, such as business analysts or data engineers. This structure contributes to a diminished sense of customer satisfaction and identification with the shared Arena Product.

However, the new Arenas can now leverage the experiences of the initial ones. Teams and System Leads from existing Arenas are transitioning to the newly created ones, serving as on-the-job mentors. System Leads facilitate lightweight exchange formats between the Arenas to maintain the flow of knowledge.

Rules

AME3 establishes Rules that foster the development and flourishing of the leadership system. Here are the rules connected with the Team.

Team

The Team is a leadership function in AME3 focused on improving customer satisfaction. Teams develop, provide, and maintain the Arena Product with stable membership. Each Team is supported by at least one System Lead.

- A Team leads to **improved customer satisfaction**.
- An Arena consists of one or multiple Teams.
- All Teams in an Arena do all the work to develop, provide, and maintain the Arena Product.
- The Team has stable membership over at least one Match, preferably for much longer.
- Each employee is a member of only one Team at a time.
- The Teams are responsible for keeping all work transparent.
- Each Team is supported by at least one System Lead.
- Team members may have different qualifications and experience. Additional roles may be defined by the Team or System Lead.

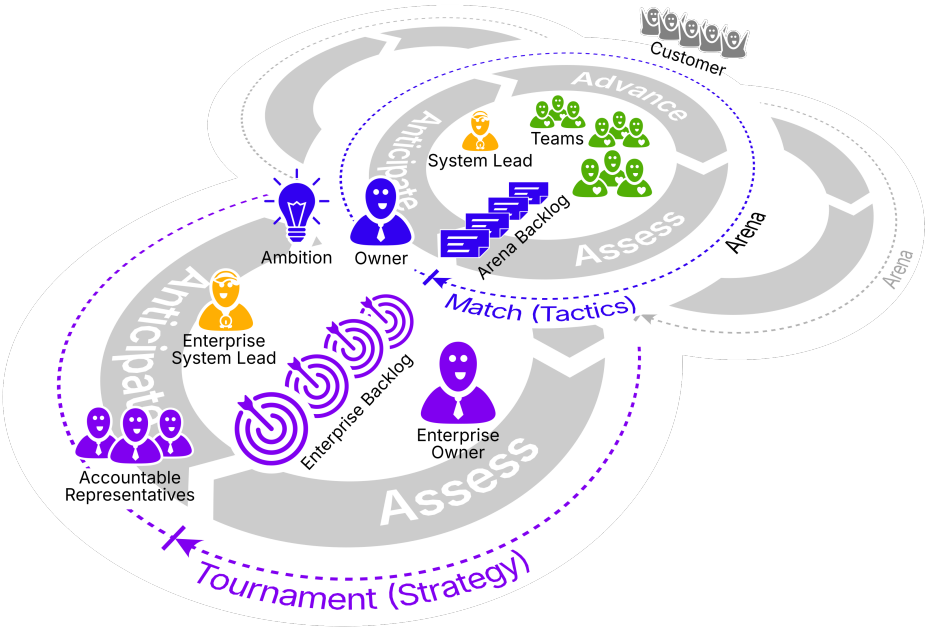
Conclusion

The Teams are leading to **customer satisfaction.** They serve the enterprise by:

- Managing and executing the work.
- Creating value and ensuring quality.
- Identifying opportunities for improvement in products, services, and work systems.

Continue reading about the meaning and functioning of AME3's Leadership System.

3. The AME3 Rules



AME3 stands for **A**daptive **M**etaframework for **E**mpirical **E**nterprise **E**volution. The framework assists an enterprise in adopting a system design grounded in three fundamental pillars: an Agile Leadership System, a Strategy for Evolution, and Enterprise-wide Rules.

The **AME3 Rules** enable companies to adapt their services and products to changing market conditions using empirical evidence. They foster the simultaneous evolution of services and products with the organizational structure.

AME3 sets a minimal yet sufficient number of rules to allow integration with other frameworks and methods. Thoroughly understand the impact of these rules on the entire system before attempting to make any changes. The framework targets primarily small to medium-sized enterprises (SMEs),

though larger enterprises can implement it within sufficiently autonomous divisions.

The AME3 Rules alone are insufficient to enhance agility and results. Enterprises should adopt other methods and frameworks like Scrum, LeSS, LeSS Huge, Lean, Kanban, Wardley Mapping, or Cynefin. The choice should align with the evolutionary stages of the product and organization and the particular Ambitions.

Although some frameworks and methods may have different rules and recommendations, the combined use outweighs the inconsistencies. The AME3 Rules support easy mapping and integration with other frameworks, enhancing overall results. See Example Mapping of Methods and Frameworks for a practical illustration.

The **AME3 Rules** organize into two sections: the **Rules of the Arena**, where teams create and evolve products and services, and the **Rules of the Enterprise**, which provides the strategic context of Arenas. Each level defines Leadership Functions, Artifacts, and Constraints.

The **AME3 Rules** by design do not delve into the reasons behind the rules, how they work, or specific usage instructions. The Leadership, Strategy, and Foundation chapters provide the theoretical background. The Playbook and Interplay chapters help with practical application.

Arena

An Arena is a highly independent organizational unit within an Enterprise dedicated to a specific Ambition. It contains a complete work system with Teams, Owner, and System Lead. The Arena is defined by its Leadership functions, Artifacts, and Constraints.

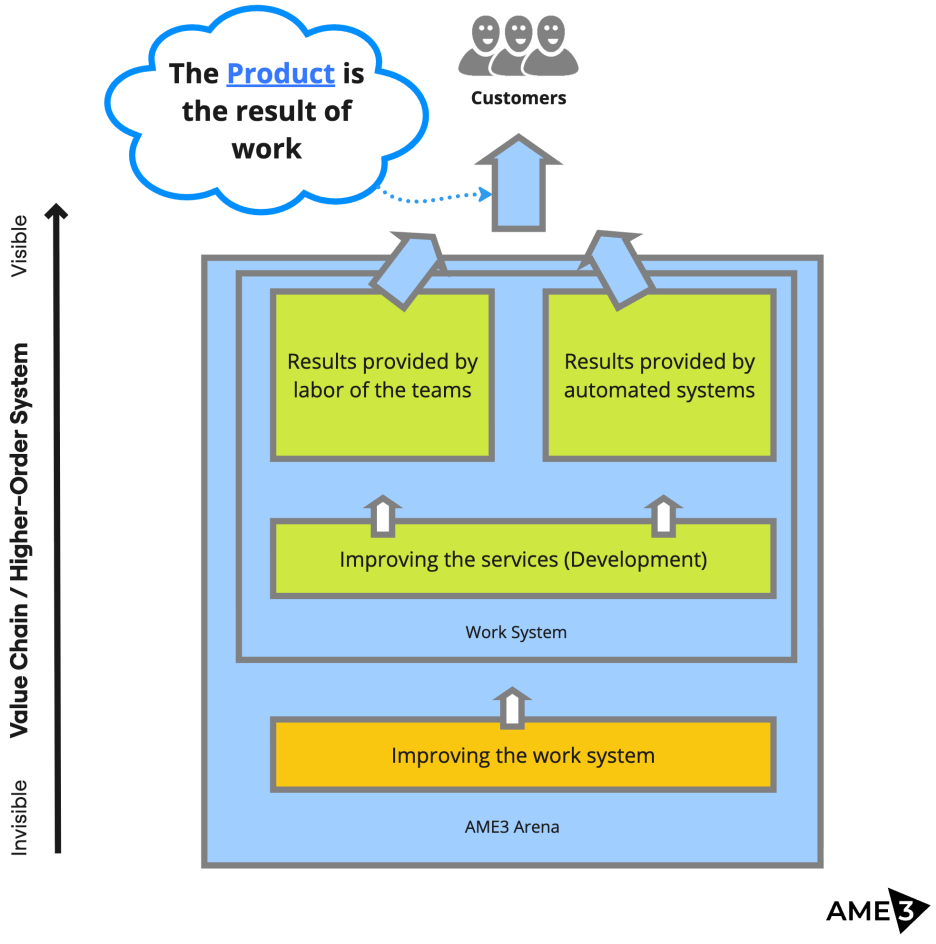
Arena Backlog

The Arena Backlog is the ordered collection of Improvements awaiting Team selection in an Arena. Teams and the Owner collaboratively create and refine these Improvements, with the Owner holding ultimate accountability for prioritization.

- The Arena Backlog is the collection of all Improvements not yet pulled by a Team.
- Improvements in the Arena Backlog are created, defined, and refined by both the Teams and the Owner.
- The Owner has the authority to order the Arena Backlog, or delegate this responsibility to the Teams, but remains ultimately accountable.

Arena Product

The Arena Product is the result of the Arena's work, encompassing services and goods delivered by Teams, product development, and work system improvements. The Product evolves with each completed Improvement.



- The Arena Product is the result of the Arena's work. It encompasses:

1. The services provided by the Teams' labor.
2. The services or goods provided by systems created by the Teams, which customers utilize or potentially use in the future.
3. The work to improve these services. Often called product development.
4. The work to improve the work system.

- The Product evolves with each Improvement completed by a Team.
- The Arena's Product is part of the Enterprise Product.

Improvement

An Improvement represents an enhancement to the Arena Product or work system. Teams pull Improvements from the Arena Backlog and commit to completing them within one Match, with Owner support for just-in-time decisions.

- An Improvement represents an enhancement to the Arena Product. This includes improvements of the work system within the Arena.
- Each Improvement can exist in one of three states: In the Arena Backlog, pulled by a Team, or done.
- Once an Improvement is pulled, a Team is expected to complete it within one Match.
- The Owner must ensure that decisions can be made just-in-time when a Team starts working on an Improvement.
- Once a Team has pulled an Improvement, it commits to focusing all its resources to complete it within the Match. If unsuccessful, the Team must identify and implement necessary improvements for the work system and/or Arena Product.

Match

The Match defines the fixed-period constraint for Arena work in AME3. It implements the Anticipate, Advance and Assess loop. Teams autonomously manage improvements within this monthly cycle. System Leads ensure effective structures are in place.

- The Match is the Anticipate, Advance and Assess Loop for the Arena.
- Each Match is a fixed period during which all work in an Arena is carried out, lasting at most a month.
- In Enterprises with multiple Arenas, all Arenas follow the same Match schedule.
- Teams are responsible for self-managing all work within a Match.
- The System Leads ensure adequate work and communication structures are in place to support the Teams throughout a Match.
- Only the Owner has the authority to cancel or restart a Match.
- Each Team is expected to complete at least one Improvement from the Arena Backlog during each Match.
- The System Leads oversee the application of Anticipate, Advance and Assess practices within the Match.

Anticipate

- Teams autonomously decide the number of Improvements they aim to complete in a Match
- A Team must pull Improvements from the top of the backlog but can discuss the order with the Owner.

Advance

- Once a Team pulls an Improvement, they are solely responsible for its completion, including coordinating with other teams and stakeholders both within and outside the Arena.
- Teams and the Owner collaboratively establish rules for completing Improvements.
- The System Leads ensure that these rules are established, adhered to, and modified as necessary to maintain efficiency and effectiveness.

Assess

- The Arena Product is regularly inspected by the Teams as part of their routine during a Match.
- At the end of each Match, Teams identify Improvements for the Product, which the Owner must then consider for inclusion in the Arena Backlog.

- At the end of each Match, Teams assess their processes to pinpoint enhancements aimed at boosting effectiveness. These enhancements are added as Improvements to the Arena Backlog.
- During the next Match's Anticipate phase, the Teams, Owner, and System Leads commit to addressing at least one improvement in the work system.

Owner

The Owner is the leadership function responsible for the success of the Arena Product in alignment with the enterprise Ambition. They hold authority over Arena Backlog ordering and capacity decisions. The Owner collaborates with System Leads while maintaining distinct accountability.

- The Owner leads to the success of the Arena Product, in line with the Ambition.
- The Owner determines how the Arena Backlog is ordered.
- Only one person serves as the Owner, who cannot simultaneously be a System Lead.
- The Owner sets the maximum number of employees in the Arena based on operational constraints.
- The Owner must consider the System Lead's recommendations.
- The Owner is an Accountable Representative.

System Lead

The System Lead ensures an effective work system within the Arena and guarantees compliance with AME3 rules. They establish and enforce additional frameworks and methods as needed. Multiple System Leads can serve an Arena, collectively accountable for all Teams.

- A System Lead leads to an effective work system in the Arena.
- A System Lead ensures compliance with the AME3 rules.
- An Arena can have multiple System Leads.
- The System Leads are entitled to establish, discard, and demand compliance with rules beyond the AME3 Framework. This includes additional frameworks and methods.
- A System Lead cannot simultaneously serve as the Owner and is appointed by the Owner.

- A System Lead may specialize in certain Teams, yet collectively all System Leads remain accountable for all Teams.

Team

The Team is a leadership function in AME3 focused on improving customer satisfaction. Teams develop, provide, and maintain the Arena Product with stable membership. Each Team is supported by at least one System Lead.

- A Team leads to **improved customer satisfaction**.
- An Arena consists of one or multiple Teams.
- All Teams in an Arena do all the work to develop, provide, and maintain the Arena Product.
- The Team has stable membership over at least one Match, preferably for much longer.
- Each employee is a member of only one Team at a time.
- The Teams are responsible for keeping all work transparent.
- Each Team is supported by at least one System Lead.
- Team members may have different qualifications and experience. Additional roles may be defined by the Team or System Lead.

Enterprise

The Enterprise defines the organizational entity in AME3 that provides resources and support for Arenas. It establishes clear boundaries through Leadership Functions, Artifacts, and Constraints to ensure Arena autonomy and independence.

- The Enterprise provides resources to support an Arena.
- An Enterprise may encompass multiple Arenas.
- If the Enterprise operates other business areas with products and services outside the AME3 framework, it ensures these do not interfere with the independence of the Arenas.
- The boundaries of an Enterprise are defined by its Leadership Functions, Artifacts, and Constraints.

Ambition

The Ambition defines the purpose, expected successes, and constraints for each Arena Product in the enterprise. It justifies an Arena's existence and guides the Owner's leadership. Annual refinement ensures alignment and commitment across all Team members.

- The Ambition clarifies the purpose, expected successes, and constraints associated with the Arena Product, including financial limitations.
- The Ambition justifies the existence and operations of an Arena.
- The Owner is responsible for leading the development of the Arena Product towards achieving the Ambition.
- If the Ambition is at risk, the Owner must take appropriate actions, which may include terminating the Arena Product.
- The Ambition is reviewed and refined at least annually, with discussions involving members of all Teams to align understanding and commitment.

Enterprise Backlog

The Enterprise Backlog contains all strategic Goals awaiting Arena focus. The Enterprise Owner orders this backlog, while Owners collectively define and refine Goals. Owners pull the highest priority Goals for their Arenas.

- The Enterprise Backlog is the list of all strategic Goals an Arena is not focused on or had not yet completed.
- Goals in the Enterprise Backlog are created, defined, and refined by all Owners and the Enterprise Owner. Others may also be involved.
- The Enterprise Owner has the authority to order the Enterprise Backlog, or delegate this responsibility to all Owners collectively, but remains ultimately accountable.
- Owners are obliged to pull Goals with the highest order from the Enterprise Backlog.

Enterprise Product

The Enterprise Product represents all products and services offered by the enterprise. It combines the Arena Products from all Arenas plus results

from units not yet operating within AME3.

- The Enterprise Product represents the combined result of work of the entire Enterprise. It includes the Products from all Arenas and results from all other units not yet operating within AME3.
- The term 'Enterprise Product' is synonymous with the phrase 'all products and services offered or are in preparation to be offered by the enterprise'.

Goal

The Goal is a strategic objective that provides direction for all Teams within an Arena. Set by the Owner, it defines what needs to be achieved within one to nine Matches. The Goal ensures alignment between Arena work and Enterprise Ambitions.

- The Goal is a strategic objective that guides all Teams within an Arena.
- Among others, a Goal can be a new Arena Product, the initiation or reorganizing of an Arena, or changes to an Ambition.
- Multiple Arenas can share the same Goal.
- The Goal is moved into focus by the Owner of the Arena and at the transition between Matches.
- The Owner can remove or replace a Goal at will.
- The Owner, System Lead, and Teams are collectively responsible for aligning their efforts towards achieving the Goal.
- The Owner must ensure that there is always at least one Goal in focus.
- There may be more than one Goal in focus, but the Owner is encouraged to keep the number to a minimum.
- A Goal should encompass the scope of work for all Teams for at least one Match but should not exceed nine Matches in duration.
- The Owner must ensure that the Goal aligns with the Ambition.
- The Owner must ensure that the Goal and the Arena Backlog are well-aligned.
- Each Goal can exist in one of three states: in Enterprise Backlog, In focus, or completed.

Tournament

The Tournament is the enterprise-level constraint that implements the Anticipate, Advance and Assess loop across the entire organization. It spans one or multiple Matches and enables Accountable Representatives to assess the Enterprise Product, while the Enterprise Owner decides on strategic changes to Ambitions and the Enterprise Backlog.

- The Tournament is an Anticipate, Advance and Assess Loop for the entire Enterprise.
- Each Tournament is a fixed period of one Match or a multiple of Matches, but not more than one year, preferably 3 months.
- With each Tournament the Accountable Representatives …
    1. … assess the Enterprise Product by updating their landscape toward evolution, and
    2. … suggest changes to Ambitions and the Enterprise Backlog, and
    3. the Enterprise Owner decides on changes to the Ambitions and the Enterprise Backlog.
- The Enterprise System Lead oversees the application of Anticipate, Advance and Assess practices within the Tournament. Planning and Execution are covered by the Matches of the Arenas.

Accountable Representative

The Accountable Representative holds ultimate accountability for Enterprise success and societal impact. They have collective authority to initiate or discontinue Arena Products while respecting Arena autonomy. They receive full transparency and generate strategic insights.

- An Accountable Representative is accountable for the overall success of the Enterprise.
- An Accountable Representative is accountable to the society for the actions of the Enterprise.
- The Accountable Representatives have the collective authority to initiate or discontinue any Arena Product and its corresponding Arena.
- Accountable Representatives receive complete transparency regarding all artifacts within an Arena.

- They are tasked with generating additional insights as needed, utilizing data from Arena artifacts.
- While Accountable Representatives can suggest modifications, they cannot mandate changes to the Artifacts, Functions, or Constraints of an Arena.
- They may suggest Improvements for the Arena Backlog or Goals of the Enterprise Backlog but cannot override the decisions of the Enterprise Owner or Owners.
- All Owners and the Enterprise Owner are Accountable Representatives.

Enterprise Owner

The Enterprise Owner is the executive leadership function responsible for the success of the Enterprise Product. They have the authority to initiate or stop Arenas, change Ambitions, and order the Enterprise Backlog while considering Enterprise System Lead recommendations.

- The Enterprise Owner leads to the success of the Enterprise Product.
- The Enterprise Owner can initiate or stop an Arena and/or its Product.
- The Enterprise Owner can change the Ambition of an Arena.
- The Enterprise Owner is an Accountable Representative.
- The Enterprise Owner cannot be a System Lead or Enterprise System Lead.
- The Enterprise Owner must consider the Enterprise System Lead's recommendations.
- The Enterprise Owner determines how the Enterprise Backlog is ordered.

Enterprise System Lead

The Enterprise System Lead is the System Lead for the enterprise-level work system in AME3. This leadership function ensures effective structures and processes across all Arenas while maintaining system coherence. They can also serve as System Lead for individual Arenas.

- The Enterprise System Lead is the System Lead for the work system on Enterprise level.

- They can also be System Lead for an Arena.

Postscript

Postscript

Applying the AME3 Rules amounts to a radical shift for most companies because it typically requires significant restructuring. For new enterprises, adopting AME3 from the start is relatively straightforward. However, existing companies may need to navigate complex pathways and undertake major restructuring efforts at varying intervals. The AME3 Rules help identify necessary organizational changes.

Unlike prescriptive scaling frameworks, AME3 allows organizations the flexibility to steadily adapt their structures and processes to meet the demands of a competitive environment. The Agile frameworks and methods recommended within AME3 may occasionally conflict in detail. In cases of discrepancy, the principles of inspection and adaptation are invaluable, which is a Strategic Doctrine in AME3.

Part II.

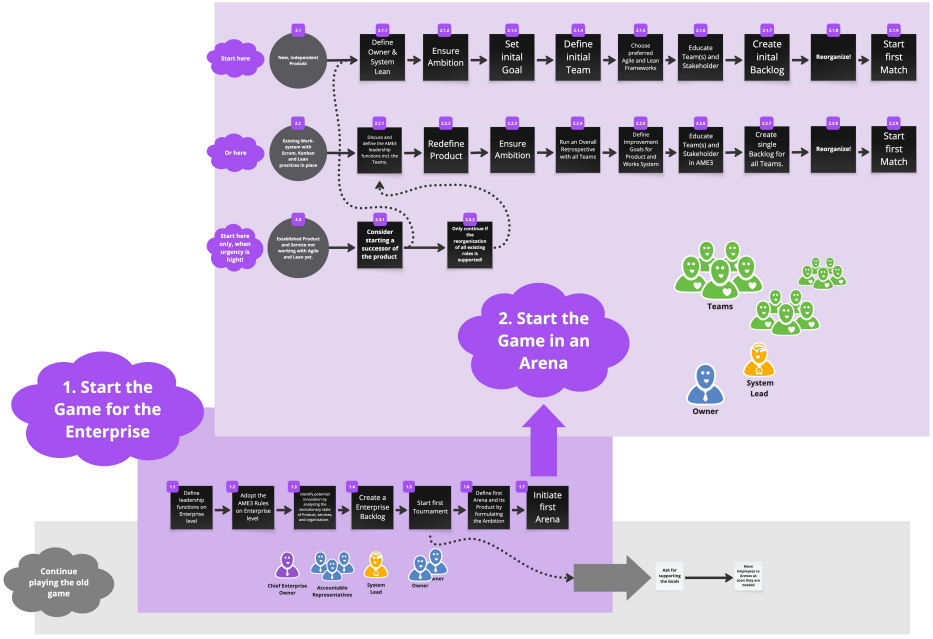
Playbook

Adopting AME3 does not require a large-scale organizational transformation. You don't need to reorganize everything at once. Instead, AME3 provides a structured path: start with lightweight enterprise-level governance, then grow one Arena at a time.

First, start the game for the enterprise — establish the leadership functions, Rules, and Strategy that provide strategic direction. Then, start the game in an Arena — set up the tactical work system where Teams deliver and evolve products and services. Each Arena has its own entry point depending on where you stand: a brand-new product, an existing agile team adopting AME3 as its guiding framework, or an established organization starting its evolution.

Pick the entry point that fits your context and scale at your own pace.

This Playbook is deliberately practical and structured. It focuses on the *what* and *how* of adoption. If you want to see these steps come to life, the Leadership System section tells the story of two organizations following this Playbook. One starting from scratch, one evolving an existing structure.



4. Start the Game for the Enterprise

Why Start at the Enterprise Level?

Many organizations adopt agile methods and lean practices at the team level. Teams run Sprints, hold Retrospectives, and improve locally. Yet the enterprise as a whole does not evolve. Decisions still flow top-down. Strategic priorities remain unclear. Teams optimize their own work while the organization drifts.

This is sub-optimization. Without enterprise-level alignment, tactical improvements stay local. They don't compound into strategic advantage. Teams pull in different directions. Investments scatter across too many initiatives. The result: effort without impact — *Wirkung* without direction.

AME3[1] addresses this by starting at the enterprise level. Before any Arena begins its work, the enterprise establishes lightweight governance: clear leadership functions[2], shared Rules, and a Strategy[3] that guides evolution. This is not a heavy bureaucracy. It is the minimum structure needed to ensure that every Arena's work contributes to the enterprise's strategic direction.

The approach is empirical. Rather than planning a complete transformation upfront, AME3 creates a feedback loop at the enterprise level — the Tournament — where leaders inspect progress and adapt strategy based on evidence. Evolution replaces revolution.

[1] see: AME3, 185
[2] see: Leadership, 47
[3] see: Strategy, 19

The Steps

1.1 Define Leadership Functions on the Enterprise Level

Establish the enterprise-level leadership functions. The Enterprise Owner takes accountability for the success of the Enterprise Product. Accountable Representatives provide oversight and ensure alignment with the enterprise's broader obligations. The Enterprise System Lead ensures effective structures and processes across all Arenas.

These are not new management layers. They are clearly defined accountability functions that replace ambiguous decision-making with transparent leadership.

1.2 Adopt the AME3 Rules on the Enterprise Level

Adopt the Rules as the shared operating agreement for the enterprise. The Rules establish a common language and foundational practices. They define how Arenas relate to the enterprise, how strategic decisions are made, and what autonomy Arenas have.

The Rules are deliberately minimal. They provide just enough structure to enable coherent evolution without constraining how individual Arenas organize their work.

1.3 Identify Potential for Innovation

Analyze the current state of all products, services, and organizational structures. Where is the enterprise today? What is evolving well? What is stagnating? Where do market opportunities or competitive pressures demand attention?

This assessment creates the raw material for strategic prioritization. It reveals where new Arenas could create the most value and where existing work needs strategic realignment.

1.4 Create an Enterprise Backlog

Build the Enterprise Backlog — the ordered list of strategic Goals for the entire enterprise. The Enterprise Owner orders this backlog based on strategic priorities. Owners collectively define and refine Goals.

The Enterprise Backlog makes strategy visible and actionable. Every Goal represents a concrete strategic objective, not a vague aspiration. This transparency enables informed trade-offs and prevents the silent accumulation of conflicting priorities.

1.5 Start the First Tournament

Launch the first Tournament — the enterprise-level cycle of strategic inspection and adaptation. During a Tournament, Accountable Representatives assess the Enterprise Product while the Enterprise Owner decides on strategic changes to Ambitions and the Enterprise Backlog.

The Tournament establishes the empirical heartbeat of the enterprise. It is where strategy meets reality and adjusts accordingly.

1.6 Define the First Arena and Its Product

Select the first Arena by formulating an Ambition. The Ambition defines the purpose, expected successes, and constraints for an Arena Product. It justifies why this Arena should exist and provides the strategic frame for its Owner.

Choose carefully. The first Arena sets the pattern for all that follow. Pick a product or service area where AME3 can demonstrate clear value.

1.7 Initiate the First Arena

With the enterprise-level structures in place and the first Ambition defined, initiate the first Arena. Appoint an Owner and transition into the arena-level setup described in Start the Game in an Arena[4].

[4] see: Start the Game in an Arena, 89

4. Start the Game for the Enterprise

The rest of the organization continues operating as before. Employees move to Arenas as they are needed — there is no big-bang reorganization. The enterprise evolves one Arena at a time, guided by strategic Goals and governed by the Tournament cycle.

5. Start the Game in an Arena

What Is an Arena?

An Arena is a highly independent organizational unit within an Enterprise, dedicated to a specific Ambition. It contains a complete work system: Teams that deliver and improve the Arena Product, an Owner who drives product success, and a System Lead who ensures an effective work system.

Independence is essential. An Arena must be able to evolve its product without depending on other Arenas for every decision, resource, or release. This autonomy enables speed. It allows Teams to respond to customer needs and market changes without navigating enterprise-wide coordination for every step.

But independence does not mean isolation. Every Arena operates within the strategic frame set by the enterprise. The Ambition defines the Arena's purpose and constraints. Goals from the Enterprise Backlog provide strategic direction. The Tournament creates the feedback loop between Arena results and enterprise strategy.

Tactics need strategic guidance and doctrines. An Arena without enterprise alignment optimizes locally — exactly the sub-optimization AME3 is designed to prevent.

Three Entry Points

Not every Arena starts from the same place. The AME3 Playbook defines three entry points depending on the Arena's starting situation.

- **Entry Point 2.1 — New, Independent Product.** Start here when the Arena will develop a completely new product or service with no existing teams or processes in place.

- **Entry Point 2.2 — Existing Work System with Agile and Lean Practices.** Start here when the Arena takes over an existing product that already operates with Scrum, Kanban, Lean, or similar practices.
- **Entry Point 2.3 — Established Product without Agile and Lean Practices.** Start here only when urgency is high. This applies to established products and services that have not yet adopted agile or lean ways of working.

Entry Point 2.1 — New, Independent Product

This path applies when the enterprise creates a brand-new Arena for a new product or service. You start with a clean slate and full freedom to design the Arena from the ground up.

2.1.1 Define Owner and System Lead

Appoint an Owner who will be accountable for the success of the Arena Product. Appoint a System Lead who will build and maintain an effective work system. These two leadership functions form the foundation of the Arena. They must be held by different people — the separation ensures balanced decision-making between product direction and work system effectiveness.

2.1.2 Ensure Ambition

Verify that a clear Ambition exists for this Arena. The Ambition defines the purpose, expected successes, and constraints including financial boundaries. If the Ambition was already formulated during the enterprise setup (Step 1.6), review it with the newly appointed Owner and System Lead. Make sure it is understood and actionable.

2.1.3 Set Initial Goal

The Owner sets the first Goal — a strategic objective that provides direction for the Arena's initial work. The Goal defines what needs to be achieved within one to nine Matches. It translates the broad Ambition into a concrete, time-bound objective.

2.1.4 Define Initial Team

Form the first Team. A Team is a stable group with full responsibility for delivering and improving the Arena Product. Consider the skills needed to achieve the initial Goal. Keep membership stable — each employee belongs to only one Team at a time.

2.1.5 Choose Preferred Agile and Lean Frameworks

Select the agile and lean frameworks that best fit the Arena's context. AME3 does not prescribe a specific method at the Team level. The System Lead may choose Scrum, Kanban, or other approaches. The choice should serve the Arena's needs, not follow a template.

2.1.6 Educate Teams and Stakeholders

Train all Team members and relevant stakeholders on AME3 and the chosen frameworks. Everyone needs to understand the Rules, their leadership functions, and how the Arena operates within the enterprise structure.

2.1.7 Create Initial Backlog

Build the first Arena Backlog. The Owner orders the backlog based on the current Goal. The backlog makes work visible, enables prioritization, and ensures that every Improvement contributes to the Goal.

2.1.8 Reorganize

Put the new structure into effect. People move into their new roles. The Arena becomes operational. This is the moment where planning turns into action.

2.1.9 Start the First Match

Begin the first Match — the fixed monthly cycle during which all Arena work is carried out. Teams autonomously manage Improvements within this cycle. The System Lead ensures effective structures. The Owner steers product direction. The game has started.

Entry Point 2.2 — Existing Work System with Agile and Lean Practices

This path applies when the Arena takes over an existing product with teams already using Scrum, Kanban, or Lean. The challenge is transitioning established practices into the AME3 structure without losing momentum.

2.2.1 Discuss and Define AME3 Leadership Functions

Map the existing roles to AME3 leadership functions. Who becomes the Owner? Who takes the System Lead role? How do existing teams map to Teams in AME3? This step requires open discussion with everyone involved. Existing Scrum Masters, Product Owners, and managers need clarity on how their responsibilities evolve.

2.2.2 Redefine the Product

Clarify the Arena Product. In existing organizations, product boundaries are often blurry. Multiple teams may work on loosely related features without a unified product definition. Define a single, coherent Product that the Arena is accountable for.

2.2.3 Ensure Ambition

Establish or refine the Ambition for this Arena. Existing teams often operate without a clear strategic mandate. The Ambition provides that mandate — it defines why this Arena exists and what success looks like.

2.2.4 Run an Overall Retrospective with All Teams

Bring all Teams together for a joint retrospective. Assess the current state of the work system, the product, and inter-team collaboration. Identify what works, what doesn't, and what needs to change for AME3 to succeed. This creates shared understanding and buy-in.

2.2.5 Define Improvement Goals for Product and Work System

Based on the retrospective, the Owner defines Goals that address both product evolution and work system improvements. These Goals bridge the gap between where the Arena is today and where AME3 needs it to be.

2.2.6 Educate Teams and Stakeholders in AME3

Train everyone on AME3. Even experienced agile practitioners need to understand how AME3 differs from their current setup — especially the enterprise-level governance, the Tournament cycle, and the distinct accountability of Owner and System Lead.

2.2.7 Create a Single Backlog for All Teams

Consolidate all existing backlogs into one Arena Backlog. This is often a significant shift. Multiple team backlogs become a single, Owner-ordered backlog. This creates transparency and enables true prioritization across all Teams.

2.2.8 Reorganize

Implement the structural changes. Adjust team compositions if needed. Formalize the new leadership functions. Make the transition visible and official.

2.2.9 Start the First Match

Begin the first Match under AME3. The familiar cadence of agile work continues, but now within a coherent Arena structure connected to the enterprise strategy.

Entry Point 2.3 — Established Product without Agile and Lean Practices

This path is for legacy products and services that have not yet adopted agile or lean ways of working. Proceed only when urgency is high.

2.3.1 Consider Starting a Successor of the Product

Before transforming the existing organization, consider whether it is better to start a new Arena that builds a successor product (Entry Point 2.1). A successor approach avoids disrupting the existing operation and allows the new Arena to move fast without legacy constraints.

2.3.2 Only Continue If Reorganization of All Existing Roles Is Supported

If a successor is not feasible, the only option is to transition the existing organization. This requires full support from leadership for reorganizing all existing roles into AME3 leadership functions. Without this commitment, the transition will stall.

If support is confirmed, continue with either Entry Point 2.1 (treating the product as new within AME3) or Entry Point 2.2 (transitioning the existing structure).

6. Don't Trust the Playbook

This Playbook lays out a clear sequence of steps. It tells you what to do first, what comes next, and where to go from there. That is useful. It gives you orientation and a starting point. But don't mistake it for a plan you can follow blindly.

A Simplification, Not a Blueprint

Every playbook is a simplification. It takes something complex and presents it as a series of steps. That makes it accessible. It helps leaders and teams build a shared understanding of what AME3 adoption looks like and what is coming next. Potentially.

The key word is *potentially*. A playbook can show you a likely path, but it cannot predict the path your organization will actually take. Your enterprise has its own history, culture, market pressures, and people. No two organizations will walk the same path, and no sequence of steps can account for everything you will encounter along the way.

Use this Playbook as a compass, not as a GPS. It points you in the right direction. But the terrain you cross is yours to navigate.

Organizations Are Living Systems

Why can't you just follow the steps? Because an organization is not a machine. It is a living system. More like an organism than a factory.

In biology, Open Ended Evolution[1] describes systems that generate increasing complexity over time without a predefined endpoint. Life on Earth

[1] see: Open Ended Evolution, 204

works this way. It doesn't follow a plan. It evolves through variation, selection, and adaptation. Creating higher-order systems with even higher complexity along the way.

Your enterprise works the same way. Every decision, every reorganization, every new Arena you launch changes the system. People adapt. New behaviors emerge. Interactions between teams, products, and markets create outcomes that no one foresaw. The organization evolves, whether you plan for it or not. What you can do is provide guidance. This is why AME3's strategic doctrines are so important. They don't prescribe specific actions, but they shape how decisions are made across the enterprise.

This is exactly what makes a playbook unreliable as a literal guide. The steps described here assume a starting point. But by the time you complete step three, steps four through seven may need to look different than what this Playbook suggests. The system has changed. You have changed it.

Empirical Control: The Real Guide

If you can't trust the Playbook, what can you trust? The answer is the same principle that runs through all of AME3: empirical control[2].

Your organization is a Complex Adaptive System[3]. In complex environments, cause and effect can only be understood in retrospect. You cannot predict outcomes reliably. But you can create short feedback loops that allow you to inspect results and adapt your approach.

AME3 embeds this principle in the AAA-Loop[4]: Anticipate, Advance, and Assess. This loop is not just a practice for Teams within an Arena. It is the fundamental structure through which the entire enterprise evolves. And it maps directly to how you should use this Playbook.

Anticipate. Design your own path. Read the Playbook. Understand the steps and their intent. Then design your own path based on where your organization stands today. What makes sense as a first move? What can you skip or adapt? Formulate hypotheses about what will work in your context. This is not about guessing. It is about making informed decisions while accepting that you are operating under uncertainty.

[2] see: Empirical Control, 20
[3] see: Complex Adaptive System, 187
[4] see: Anticipate, Advance, Assess, 23

Advance. Move forward with your Arenas. Execute. Start the game for the enterprise[5]. Launch your first Arena. Let Teams begin their work. Gather real data. Observe what happens. Not just in the Arena, but across the organization. How do people respond? Where does friction arise? What works better than expected?

Assess. Inspect and adapt at the enterprise level. This is why the Playbook recommends starting the Tournament early. The Tournament is not an afterthought. It is the enterprise-level feedback loop where Accountable Representatives assess the Enterprise Product and the Enterprise Owner adapts the Strategy. Without it, you are flying blind. With it, you have a regular checkpoint to compare your hypotheses against reality and adjust course.

The Tournament is where the Playbook gets corrected. By your own experience.

Trust the Loop, Not the Steps

The steps in this Playbook will get you started. They represent a proven sequence based on years of observation and practice. But the real value of AME3 is not in any particular sequence of steps. It is in the system of empirical feedback loops that lets your enterprise learn and evolve continuously.

Follow the steps loosely. Trust the AAA-Loop completely. Anticipate where you want to go. Advance with courage. Assess with honesty. Then adjust. And keep evolving.

That is the game.

You may think this is odd. Writing a playbook and then telling you not to trust it. But that is exactly the point. Any framework that claims to have all the answers is not being honest about the reality of organizational change. Being your own critic is not a weakness. It is a discipline. Question the steps. Challenge the sequence. Test your assumptions against what actually happens. The organizations that evolve fastest are the ones that treat every plan, including this one, as a hypothesis worth testing.

[5] see: Start the Game for the Enterprise, 85

Part III.

Interplay

7. Artificial Intelligence

Artificial intelligence is reshaping the economy. New companies are emerging, work processes are evolving, and products and services must be reimagined.

AME3 offers the operational model for the enterprise of the future. Discover how integrating AME3 with AI can drive your enterprise's success.

AI Won't Fix Your Bureaucracy

A speculative but well-argued scenario[1] paints a grim picture: AI displaces white-collar workers at scale, payroll shrinks, consumer spending contracts, and a negative feedback loop spirals into economic crisis. The report went viral with millions of views and actually caused software stocks to drop. The argument assumes that AI will simply replace the people who currently do the work.

But what if the premise is wrong? What if AI does not actually reduce the amount of work?

As we describe in AI and the Principles of Evolution[2], every major technology that promised to simplify our lives ended up doing the opposite. We used it to build more complex systems, more rules, more processes, pushing ourselves right back to the edge of our cognitive capacity. This is not a bug. It is a law of human systems: **we always operate at the maximum complexity our tools allow.**

White-collar bureaucracy is a perfect example of this law in action.

[1] see: The 2028 Global Intelligence Crisis, 232
[2] see: AI and the Principles of Evolution, 4

The Bureaucracy Machine

Over decades, white-collar workers have built a self-reinforcing system of coordination, compliance, reporting, and approval. Each layer was added for a reason. Each new process solved a real problem at the time. But the cumulative effect is an organizational machine that consumes enormous energy while delivering diminishing value.

Digitalization made this worse, not better. When spreadsheets replaced paper, we did not reduce the number of reports. We created more. When email replaced memos, we did not communicate less. We communicated more, copying wider lists, spawning longer threads. When workflow tools automated approvals, we did not simplify the approval chain. We added more steps because the cost of each step dropped.

The pattern is consistent: lower the cost of a bureaucratic action, and you will get more bureaucratic actions.

AI is the most powerful cost reduction for bureaucratic work we have ever seen. An LLM can draft a report in seconds, summarize a hundred-page document, generate compliance checklists, and schedule meetings without human effort. The question is not whether organizations will adopt these capabilities. They already are. The question is what happens next.

The Fork

Here is where it gets interesting. AI disrupting bureaucratic overhead could go in two directions.

Path A: Simplify. Use AI as the catalyst to refactor bloated organizational systems. Eliminate unnecessary coordination layers. Stop generating reports nobody reads. Let teams own their outcomes end-to-end instead of routing every decision through three levels of approval. This is the path where AI actually delivers on the promise that digitalization broke: simpler, faster, more human organizations.

Path B: Amplify. Use AI to generate more reports, faster. Automate compliance checking so thoroughly that you can afford to add more compliance rules. Let AI assistants schedule more meetings, create more documentation, produce more dashboards. Pour gasoline on the fire.

Path B does not look like failure. It looks like progress. "We automated our entire reporting pipeline!" Yes, and now you have ten times more reports that still nobody reads.

Why Path B Is the Default

If history is any guide, most organizations will take Path B. Not because leaders are foolish, but because the system is self-reinforcing. Every layer of bureaucracy has a constituency. Every report has someone who requested it. Every approval step has a stakeholder who insisted on it. AI makes each of these cheaper to maintain, which removes the economic pressure that might otherwise force simplification.

This is the real risk of AI in the enterprise. Not that it replaces workers, but that it preserves and amplifies the very structures that should be questioned. The evolution principle predicts exactly this: available capacity gets filled.

Choosing Path A

Path A requires conscious leadership. It does not happen by default. Organizations that want to use AI for genuine simplification need to do more than deploy tools. They need to challenge the system itself.

This is where AME3[3] provides a structural answer. The framework is designed around small, empowered Teams that own their outcomes within an Arena. Bureaucratic overhead is minimized by design: fewer handoffs, shorter feedback loops, direct accountability. When you organize this way, there is simply less bureaucracy to automate.

The Evolution Focus[4] doctrine asks a different question than "How do we automate what we have?" It asks: "What should we stop doing entirely, and what should we evolve into next?" AI becomes a catalyst for evolution, not a preservative for the status quo.

[3] see: AME3, 185
[4] see: Evolution Focus, 37

The Real Disruption

The speculative crisis scenario gets one thing right: AI *is* a disruption. But a disruption is not the same as a replacement. It is a moment where the system is forced to reorganize.

White-collar work will not disappear. But the nature of that work must change. The organizations that thrive will be those that use this disruption to strip away the accumulated complexity that no longer serves them, and redirect human energy toward work that actually creates value: understanding customers, designing products, making decisions under uncertainty, collaborating across disciplines.

AI will not fix your bureaucracy. Only you can do that. AI just makes the choice unavoidable.

Developing a Strategy for the GenAI Era

Assess the Terrain

Do you remember Netscape? In 1996, they controlled nearly 80% of the web-browser market. Today, users start forgetting what a browser is, simply because browser-engines are part of our daily lives and deeply embedded in many of our devices we use.

What started as a novel and genesis idea in 1989 ended up being a commodity today. When you read this article, you most likely look at a Chromium, WebKit, or Gecko engine. Browser engines, as a fundamental component of the World Wide Web, have reshaped nearly every business and the way we handle knowledge today. The most valuable enterprises today would not exist without them.

This is evolution.

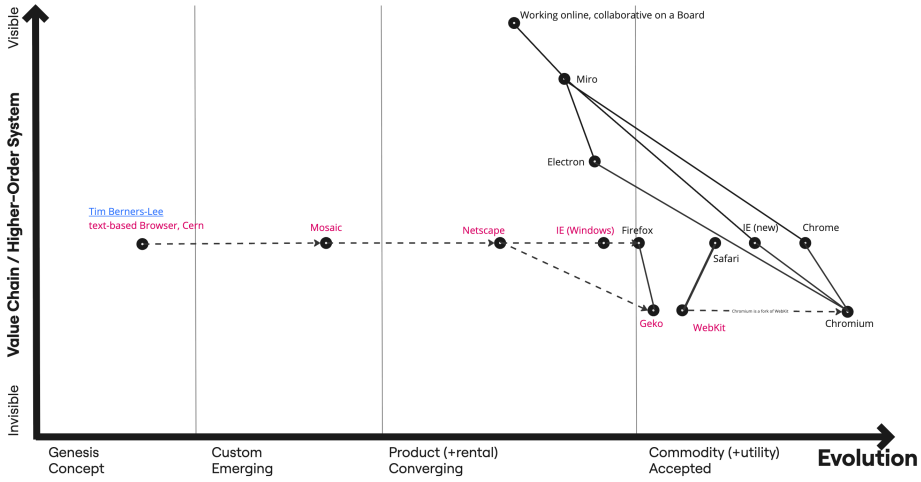


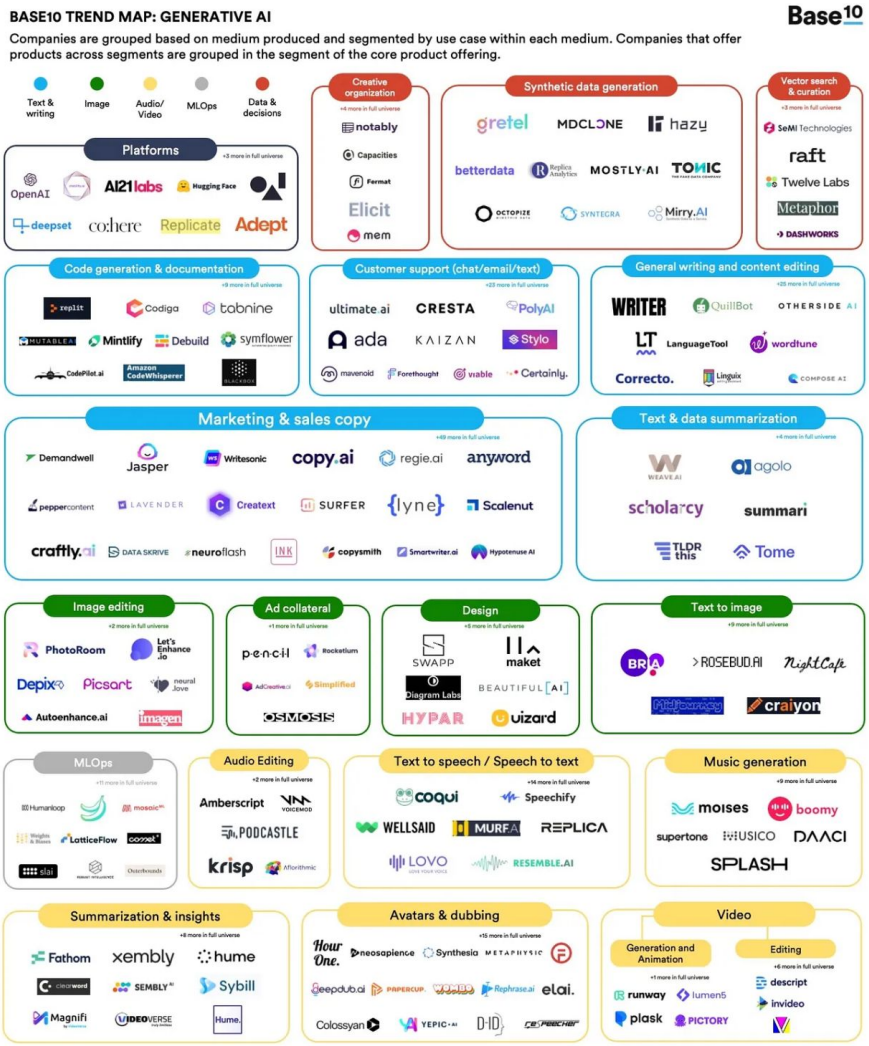
Figure 7.1.: Wardley Map of the Evolution of Web Browser. Nodes in red are historical.

GenAI will follow a similar path in its evolution. Competition demands this. But even though we know that the impact of GenAI will be as huge as with the World Wide Web, we can't foresee how it will change our business and life for sure. We are now very much at the same stage in product evolution as we were with the Mosaic browser in the 1990s. We are just starting to understand what is possible.

But what we can do now is develop a strategy for this uncertainty and new upcoming opportunities to make your enterprise fit for the GenAI age.

> The overarching strategic doctrine for an enterprise should be: Evolution Focus

I recently (January 2025) came across one of many illustrations showcasing a vast array of tools and services related to GenAI. From big names like OpenAI to fundamental services like Hugging Face. New stars like DeepSeek are still missing.



OK, it's likely that in five years, many of these names will fade from memory, much like Netscape. Other significant players of the future are probably not yet visible in the current landscape. But assessing your terrain of business in the context of GenAI tools and practices seems like a good place to start.

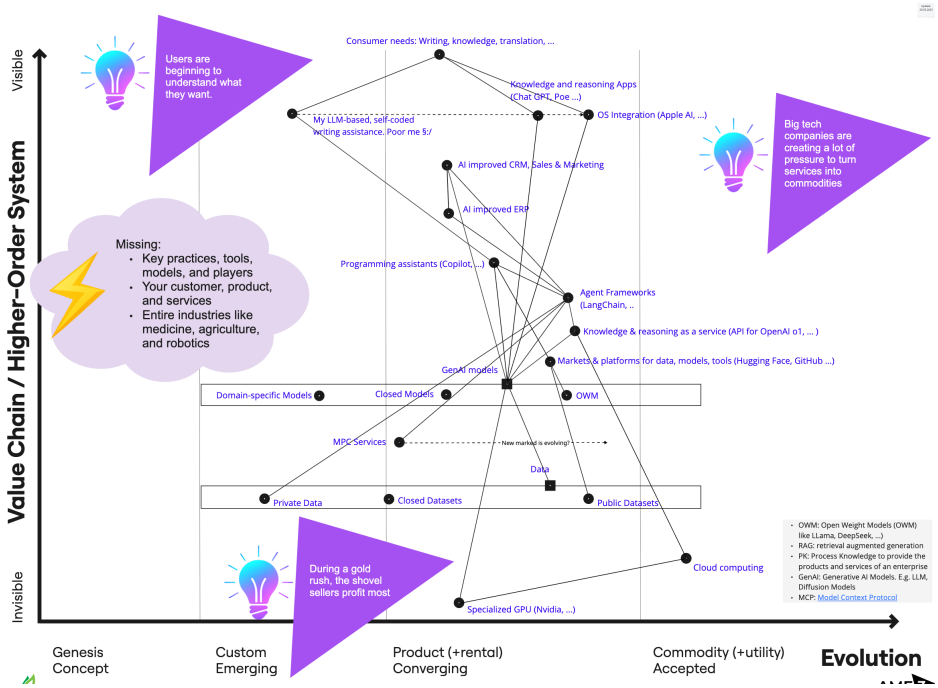
Unfortunately, illustrations that merely list the numerous GenAI tools in the market are entirely useless for making strategic decisions for your business. We need a map that sets your organization, product, and services in

the context of evolution and the GenAI market.

Thankfully, Simon Wardley created an excellent mapping technique[5] specifically for this purpose. The map of the Evolution of Web Browser I showed before is such a Wardley Map.

Besides the horizontal flow of evolution, the map shows the components like the application framework Electron relative to the value chain (y-axis). This is an essential second dimension. It shows us the dependencies from higher-order components (e.g., services, tools, practices, …) to lower, fundamental ones.

Here is my current, possibly biased and limited perspective on the GenAI market and potential forces. This is illustrated on a Wardley Map[6].



It is now easy to see why Nvidia has become so valuable. Their chips are products they can sell per piece, and everyone depends on them. Even the hyperscalers like Google, Amazon, and Microsoft still need to buy them. Patents, specialized knowledge, and the fact that the model architectures

[5] see: Wardley Mapping, 217
[6] see: Wardley Maps, 215

have been optimized for the unique chip designs are preventing competitors from keeping up with Nvidia.

However, we will see how long this advantage lasts. Optimized chip design for GenAI is definitely a candidate to become a commodity soon, much like how jeans in the Gold Rush started to become commodities. But first, it made Levi's a giant in the supplier industry.

It is also easy to see that GenAI models are the fundamental component and logic for the tech giants' push in the direction of commodity. That's why especially smaller companies and institutes release their models as open-weight models (which are not entirely open source, simply because the training data and the algorithms that learn the models are closed).

Still, GenAI models in general are far away from being commodities. As the uprising of DeepSeek has shown, new approaches and algorithms like reasoning are creating competition and opening potential for entirely new, higher-level services to customers. Now think about how everything would change if a highly optimized photonic chip, as described in this paper[7], moved into the GenAI landscape. And there is a strategy behind why OpenAI has hired (or better acquired) Jony Ive, the former chief designer of Apple.

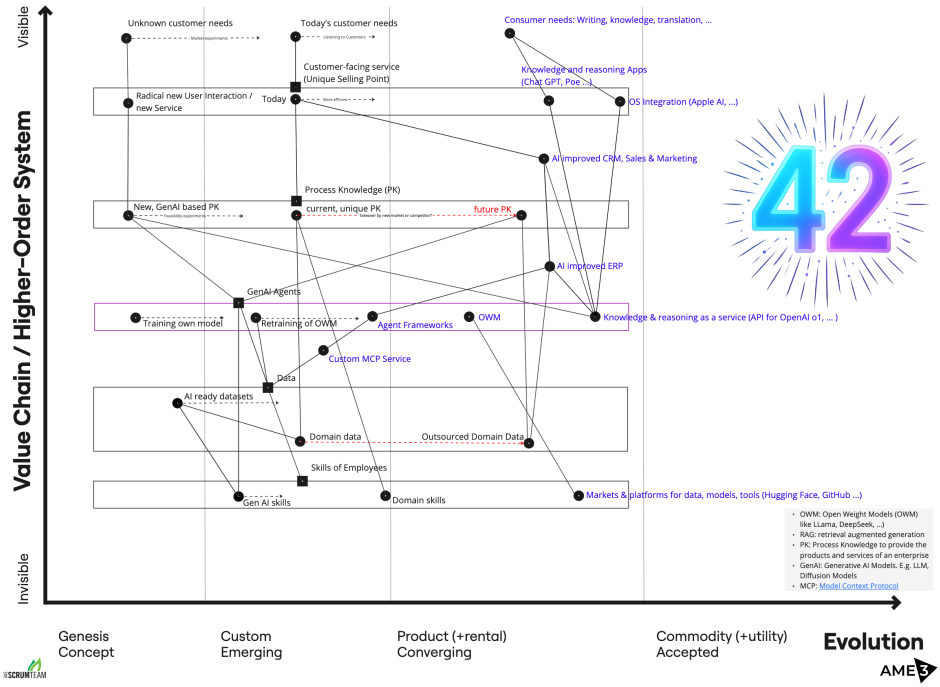
The Question about Everything …

My map of the GenAI market still poses several challenges:

1. It is probably missing positional practices, tools, models, and players you should be considering.
2. It completely lacks your customer, product, and service.
3. Entire industries are not considered: E.g., medicine development, agriculture, robotics, …

O.K., let's change that and create a map for everything. Look at the map. Then, continue reading.

[7] see: Field-programmable Photonic Nonlinearity, 196



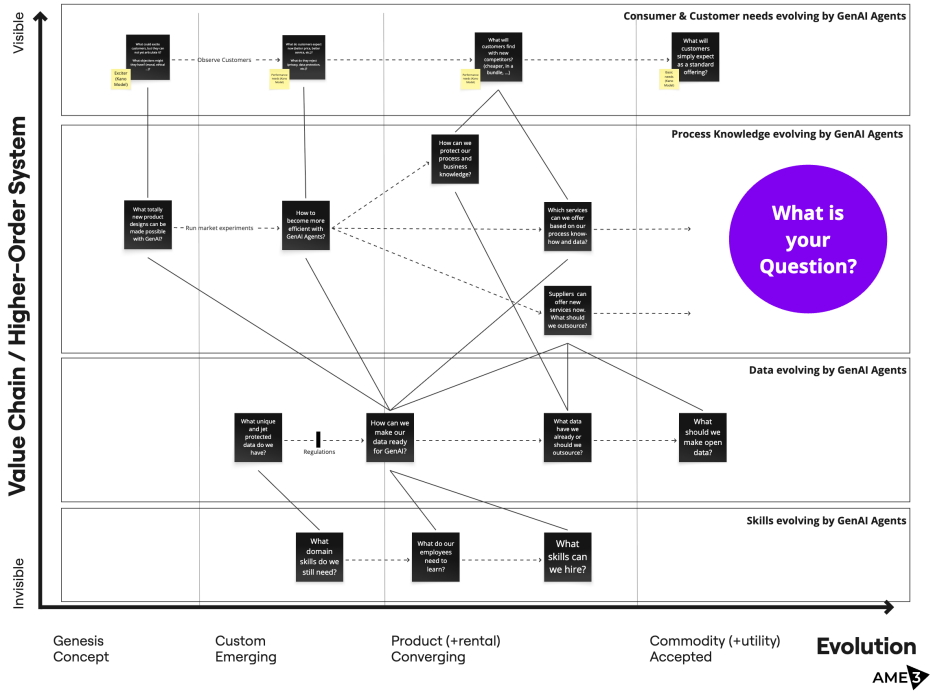
You see, it is not your map. Even if you are familiar with Wardley Maps, you probably don't understand my intention and thoughts behind it. To be honest, I don't fully grasp it either. I come up with new ideas and insights every time I look at the map and update it.

Furthermore, the true value emerges when creating maps collaboratively with others. While maps are inherently incomplete, they abstract connections within a system, enabling effective discussions.

If you want the answer to the question about the universe, life, and everything, you get 42.

So, for a more profound understanding of your terrain, you need to define the questions you would like to be answered.

Fortunately, we can extract some more specific questions from the map. I have brought them into a map again.



I structured the map into four sections along the value chain. In each section, GenAI will drive evolution forward. However, this structure is mainly to improve readability. There are no strict boundaries, and you may find a structure that fits your business better.

1. Consumer & Customer Needs

- **Genesis:** What could excite customers, but they can't yet articulate it? What objections might they have? (persona, jobs, pain points)
- **Custom:** What do customers expect now? (better price, faster service, convenience). What do they reject? (privacy concerns, data-protection issues, algorithmic bias)
- **Product (+Rental):** What can (new) competitors offer now?
- **Commodity (+Utility):** What will customers simply expect as a standard offering?

2. Process Knowledge

- **Genesis:** What entirely new product or service designs can be made possible with GenAI?
- **Custom:** How can we become more efficient with GenAI agents?

- **Product (+Rental):** How can we protect our proprietary process and business knowledge? Which services can we offer based on our process know-how and data? Suppliers can now offer new services—what should we outsource?

3. Data

- **Genesis:** What unique and yet protected data do we have?
- **Custom:** How can we make our data ready for GenAI?
- **Product (+Rental):** What data do we already own, and what should we consider outsourcing?
- **Commodity (+Utility):** What should we publish as open data to foster ecosystem growth?

4. Skills

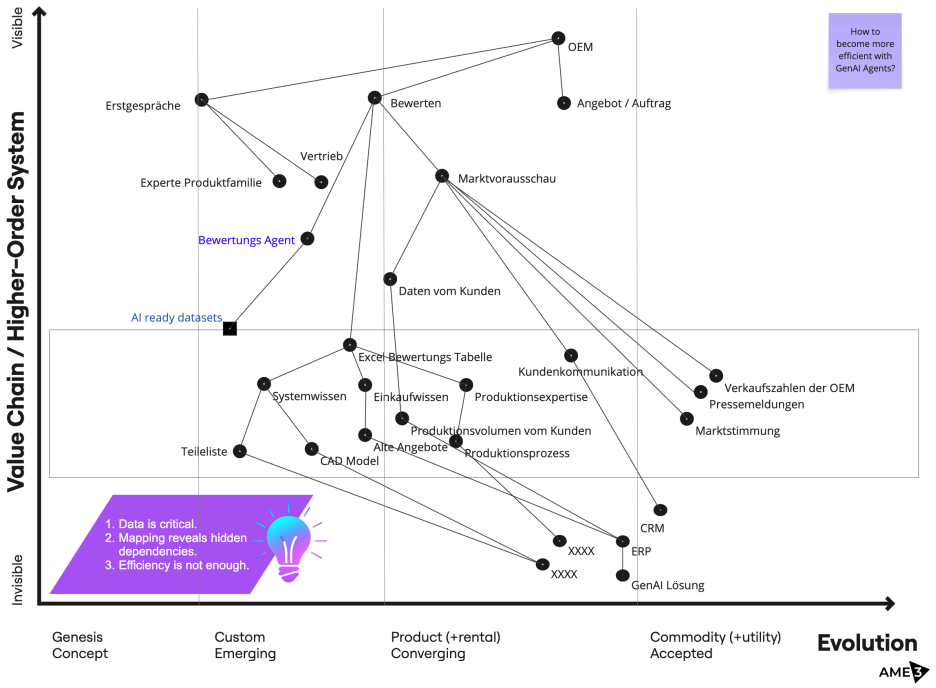
- **Genesis:** What domain-specific skills do we still need internally?
- **Custom:** What do our existing employees need to learn?
- **Product (+Rental):** What skills can we hire or contract from the market?

I hope you find inspiration from them. Choose one of the questions your business might need answers to first. Start with that.

Let's check some examples of how that can look.

Example 1: Automotive Supplier

In this example, a supplier in the automotive industry tried to understand how they could make the evaluation step in the order process more efficient and meaningful.



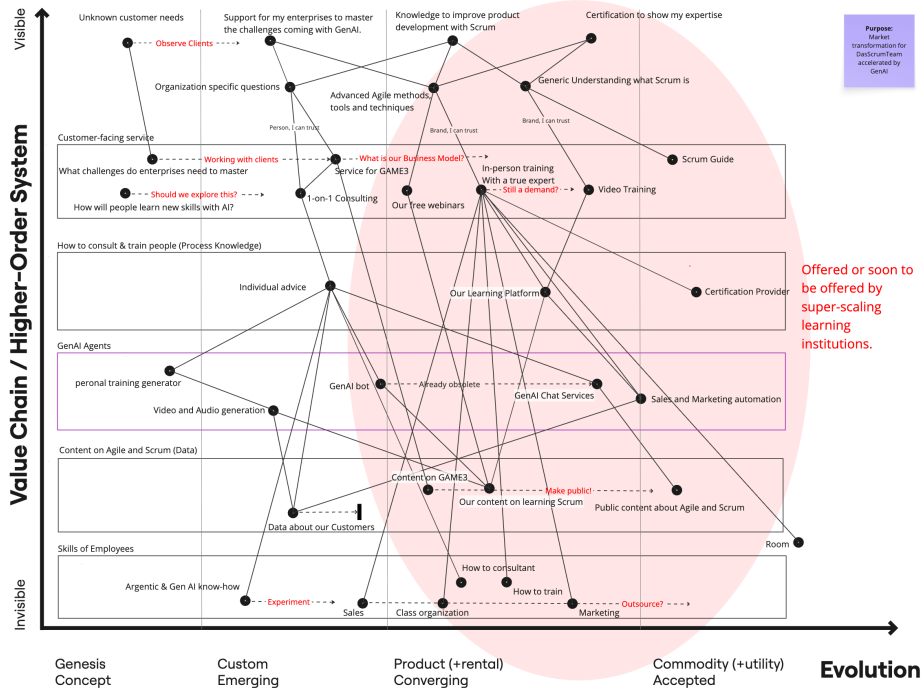
Looking at the map, it becomes pretty clear that everything depends heavily on the data they have. Some data sources are external and not yet structured. So, a first strategic goal could be to experiment and gather some data sources into datasets that an AI-agent can use.

Increasing efficiency and the quality of services is what most businesses focus on first when they bring GenAI into focus. The reason might be that this is quite obvious, as the example shows. Even though this is not a bad optimization goal, it could be a short-sighted decision because you might miss some important aspects of the market.

Example 2: Market Transformation for Scrum-Training and Certification Driven by GenAI

DasScrumTeam AG is a training and consulting boutique specialized in helping organizations utilize Scrum to improve their product development. The uniqueness of DasScrumTeam is that it combines long experience in the field of Agile with the expertise to conduct in-person training classes.

The question now is, how will GenAI impact the market for DasScrumTeam? Since the company is small and is not intending to scale massively, the conclusion is that most of the business will soon, or is already being, taken over by highly scalable training institutes like the International University. They would have had that anyway. Now just be much faster due to GenAI. For example, Duolingo just doubled the amount of online classes with GenAI in one year. Before that, it took them 10 years for the same amount.



You might be thinking now, if it was clear that this was happening anyway, why didn't the Product Owner see it coming five years ago? He did, but he was not taking action. Because he did not believe it was happening so quickly, and the company was very successful with what they were doing. I know that because I was the Product Owner. Simon Wardley calls this pattern inertia[8].

However, there are plenty of opportunities on the left side of the map. Since the company's goal is to remain boutique, here are the strategic investments.

[8]see: Climatic Patterns by Simon Wardley, 190

The Answer to Life and Everything ...

At first, it may seem that evolution is a far-fetched concept to set as the fundamental strategic doctrine for an enterprise. But what is an enterprise? It is a belief system created by humans, who are the product of billions of years of evolution.

In our examples, the future development of enterprises depends on lower-order systems. Since GenAI is a new technology bridging process knowledge and data — which most enterprises highly depend on — it will accelerate evolution by bringing more order into these areas. In other words, it reduces local entropy. This requires energy, which is very close to what we understand life to be[9].

So, Generative Artificial Intelligence[10] (GenAI) is not Artificial Life[11]. But GenAI will become part of our lives.

You are probably thinking, *"Nice philosophical thought, but how do we put these insights into action?"*

This is where the next fundamental strategic doctrine comes in: Empirical Control[12]. The Anticipate, Advance, Assess[13] Loops in AME3 are embedding this doctrine into the organization.

[9] see: What is the Purpose of Life?, 202
[10] see: Generative Artificial Intelligence, 197
[11] see: Artificial Life, 186
[12] see: Empirical Control, 20
[13] see: Anticipate, Advance, Assess, 23

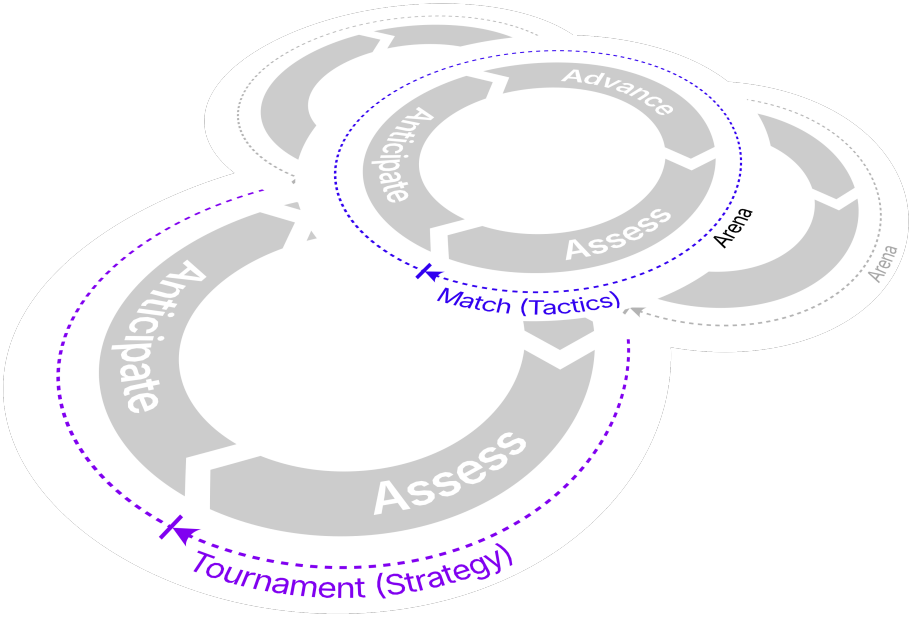


Figure 7.2.: The AAA-Loops in AME3

Next Step!

As the examples have shown, assessing your terrain by mapping the evolving GenAI landscape is only the first step. By asking the right questions and visualizing your position, you create a strong foundation for strategic decision-making. But this can only be the first step. You need to anticipate, advance, and assess again to iteratively adapt your strategy.

The AI-Enhanced Team of the Future

Do we still need development teams now that AI agents can write, test, and deploy code? The answer is more surprising than you might expect. And not a simple yes or no.

Vibe programming and agentic AI let a single engineer coordinate dozens of specialized code agents to ship production features (→ Claude Flow[14]).

Even those new to the field can achieve impressive results. For example, my daughter, a design student with no previous coding experience, built a complete Mac desktop app in just two weeks—including the build setup, version control, and GitHub page.

Critics might argue that code quality is lacking and that a junior developer may overlook important details. I heard similar concerns from C developers when I began using managed code runtimes like Java or .Net. However, the impact and scale of this shift with AI are even greater—there is no doubt about that.

So the question is natural: if AI can build most things, what is the role of a development team?

Widening the Value Chain with AI

Organizing in close groups of 7 ± 2[15] people is a pattern that most likely can be dated back to early Homo sapiens. A team of this size provides significant advantages, particularly when navigating uncertainty and complex challenges.

- **Cognitive load / communication overhead**: Every extra team member adds more potential communication lines (if everyone interacts with everyone). With too many members, coordinating and keeping everyone aligned becomes harder.
- **Accountability & motivation**: In smaller teams, each person's input is more visible; as teams grow large, individuals may feel less accountable (social loafing).
- **Skill coverage vs. coordination cost**: With fewer than ~5 people you may miss skills or people become overloaded; with more than ~9 you may get diminishing returns because the overhead of coordination overwhelms the added capacity.
- **Working memory / chunking metaphor**: The idea of "7 ± 2" from Miller is about how many separate things a person can keep in

[14] see: Claude Flow, 200
[15] see: The Magical Number Seven Plus or Minus Two, 214

mind; by analogy, a team of ~7 is still manageable in terms of keeping track of members, tasks, roles, interaction patterns.
- **Decision-making & interaction efficiency**: Smaller teams can meet, decide, adapt faster; larger ones may slow down because scheduling, alignment, consensus become harder.

GenAI accelerates the evolution of products and services. However, previous tools and technologies have also driven similar changes. Yet, none of them fundamentally altered the social pattern of the *team*. So, why should AI change this now?

One argument is that, for many, AI appears almost human. However, anyone who has worked with LLMs or diffusion models quickly sees that, while these systems are impressive at predicting human behavior, they have clear limitations. They do not replace human counterparts; instead, we use them as tools to deliver products and services for people.

When we want products and services to change, it is because humans desire it. A system that mimics our behavior can support this, but it is always guided by human intent. AI helps simplify tasks to a remarkable extent, but it remains a tool directed by people.

Looking back at past technological revolutions, we can observe that whenever we created tools to simplify work, we immediately used them to tackle greater complexity. AI is *just* one such tool. It increases throughput and knowledge access, thereby expanding the scope of the value chain that a team can manage.

Naturally, an Enterprise seeks to manage all stages of the evolution flow, either by outsourcing to suppliers or by delegating to smaller social subsystems. The size and structure of these smaller social systems depend on their specific purpose, as humans tend to organize themselves accordingly. AME3 is designed to reflect this natural order for human systems:

- Medium Enterprise (50–5,000 people): covers the entire flow from genesis to commodity. An Enterprise provides commodity services to an Arena and delegates everything that is not commodity to the Arena.
- Arena (7–250 people): a highly independent business unit. It controls the value chain to deliver converging services and products to customers. It delegates genesis and emerging tasks to Teams and provides the necessary support for them.

- Team (5–9 people): primarily focused on genesis to custom/emerging tasks. Teams discover new ways to build and deliver to customers while establishing an initial stable process. They utilize co-creative approaches like pairing.
- Pair[16] (2 people): focuses on genesis and concept work, especially when tasks are highly novel or urgent.

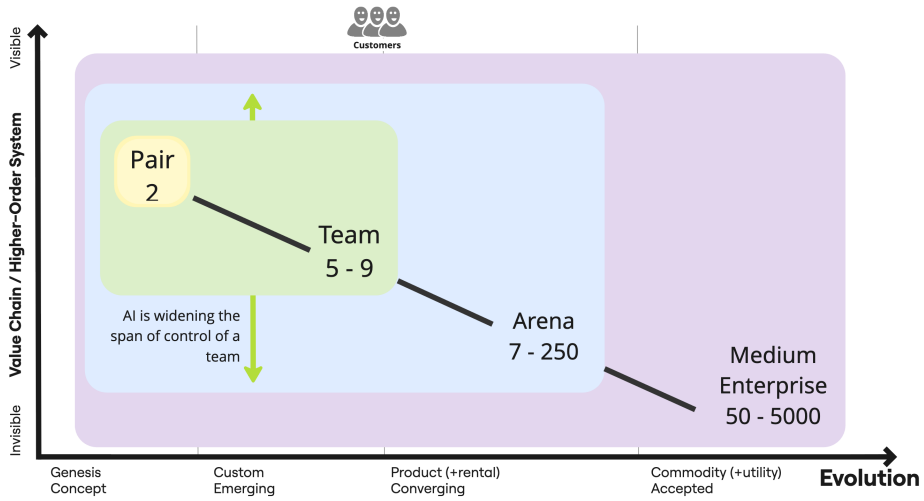


Figure 7.3.: Focus of social systems across the evolution flow in AME3

Teams are fundamental social systems. AI won't change this.

So ultimately, it is the **Teams who drive progress and deliver value** in an Enterprise. This is unlikely to change, as long as enterprises continue to seek progress and deliver value. However, the composition of teams will inevitably evolve.

How Teams Are Evolving

The pattern of team evolution over the decades is clear: fewer handoffs, faster feedback, and broader responsibility. AI will accelerate the next stage of this transformation. Let's walk through this evolution to understand it more clearly.

[16] see: Pair Programming, 220

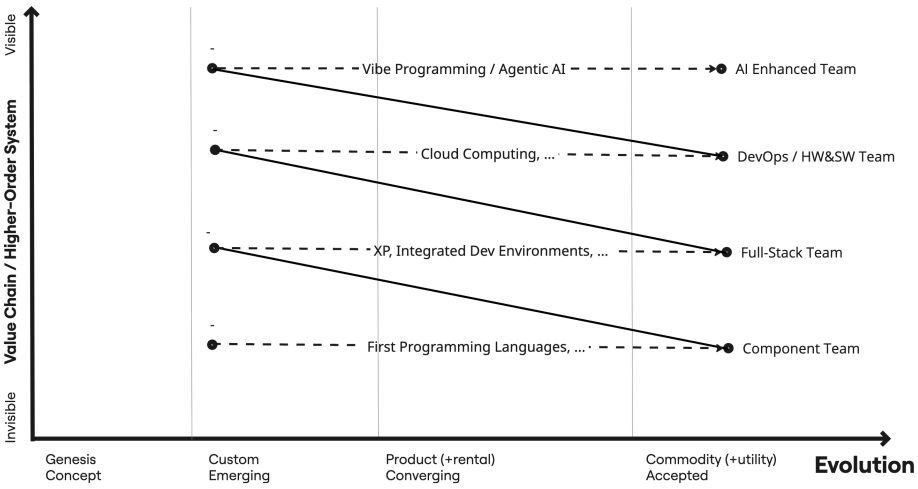


Figure 7.4.: How teams have evolved and are continuing to evolve

Era I — Component Teams

Enabled by the first programming languages.

- Work focus: isolated components and libraries; many handoffs to integrators.
- Constraints: limited tooling and automation; high coordination cost; long feedback cycles.
- Outcomes: quality inside components, but brittle integration.

Era II — Full-Stack Teams

Enabled by XP, integrated dev environments, and test automation.

- Work focus: vertical slices from UI to persistence; trunk-based flow; continuous testing.
- Constraints lifted: faster feedback; tighter product–engineering loop; fewer handoffs.
- Outcomes: shorter cycle time; improved responsibility and product fit.

Era III — DevOps / HW&SW Teams

Enabled by cloud computing and platformization.

- Work focus: build and run; infrastructure as code; observability; security by design.
- Constraints lifted: provisioning and deployment become utilities; scale and reliability increase.
- Outcomes: continuous delivery and on-call responsibility, including hardware where relevant.

Era IV — AI-Enhanced Teams

Enabled by vibe programming and agentic AI.

- Work focus: end-to-end business outcomes across customer demand, design, system architecture, service improvements, operations, and market growth.
- Constraints lifted: generation, integration, and orchestration shift to AI agents; humans focus on intent, constraints, and verification.
- Outcomes: teams are responsible for complete business capabilities and micro-enterprises, with AI as force multipliers.

  The most appropriate term for these teams is still discussed. We suggest *AI-Enhanced Team* as an initial option. Alternatively, *Full Business Teams* may be a more consistent description. We welcome your suggestions if you think there is a better term.

In pure software and IT-Areas, these teams will become *Full Business Teams*, as they can easily achieve many end-to-end business outcomes within a single cadence. In other domains, even with AI, this level of integration may not be possible. For example, mass production and product development might still require separate teams. However, fewer specialized areas will be necessary, handoffs will be reduced, and cycle times will be shortened.

Full Business Teams existed even before the rise of AI. For example, the team I led as Product Owner for seven years operated in this manner. We were a single-Team company, running very lean and outsourcing most commodity functions. Working in this way required special and radical

conditions, making *Full Business Teams* quite rare—especially in larger organizations. Our hypothesis is that this model will soon become the new normal.

New AI Products and Services

You may have noticed that the above team evolution model does not fully address a key aspect of GenAI: its potential to enable entirely new products and services, or to drive major enhancements to existing offerings. We examine this important dimension in greater depth here: *Explore how GenAI unlocks product and service innovation*[17].

What Changes for Humans?

This is likely the biggest question your employees, colleagues, and you are asking right now. I am asking it myself as well: What changes for us as humans? As with any new technology, it is difficult to answer this in detail right away.

Reflect on your experience from previous technology revolutions. GenAI has the potential to create an impact as significant as the internet (IP protocol, WWW, etc.). We can expect many changes to our current beliefs and ways of working.

What we know so far:

- **Intent and constraints:** We will focus more on defining what to build and what to avoid.
- **Taste and ethics:** We will set standards for quality and ethical boundaries.
- **Evaluation and acceptance:** We will increasingly define tests, guardrails, and meaningful measures.

The craft is moving up the stack. People will spend less time on integration and more time shaping impactful outcomes.

[17] see: Developing a Strategy for the GenAI Era, 106

The Transition Playbook with AME3

A practical flow to move from today's model of work to AI-enhanced teams:

1. Map the landscape using Wardley Mapping to identify the next goal you want to achieve, leveraging the potential of GenAI. Start with asking one of the Questions in *Developing a Strategy for the GenAI Era*.
2. Begin with a single Team or even a pair of specialists. Run experiments to validate your hypothesis.
3. If you decide to continue with your goal, continue with the same team but be prepared to change its composition. Strengthen your data and guardrails. Enhance data quality, observability, and develop security policies to ensure AI agents can operate safely.
4. Pilot a single product or service capability with a strong emphasis on delivering customer value. Choose a focused scope—such as a specific feature or internal service—leveraging an agentic toolchain.
5. Measure value delivered and customer signals.
6. Scale beyond the team. Leverage your team's expertise in GenAI technologies, customer experience, and modern collaboration methods. Expand to multiple teams within an Arena, or even launch a new Arena that operates highly independently from the rest of the organization to scale faster and adapt more swiftly to changing markets and AI advancements.

All of this must take place within a lean and adaptable governance structure at the enterprise level. Ensure that new teams are integrated into the strategic empirical control loop of AME3.

Conclusion

So, what is the answer to the question, *"Do we still need development teams now that AI agents can write, test, and deploy code?"*

If you define *development* as coding, then yes, we will see these types of teams disappear. If you define *development* as creating next-generation products and services, then this has always been—and will continue to be—the domain of teams.

AI does not remove the need for teams. It changes what teams take responsibility for and how they deliver.

Focusing only on software-based products and services—such as coding and administration—where the most visible changes are occurring, would be a major mistake. AI now offers the opportunity to break down long-standing barriers between R&D, IT, engineering, and business functions. Failing to embrace this shift will put your organization at a significant competitive disadvantage.

In AME3 terms, teams widen their scope across the value chain and become responsible for more end-to-end business capabilities. Humans set direction and standards. Agents accelerate delivery and integration.

The future team is not smaller. It is broader, more empirical, and closer to the customer—and it wins by learning faster than the market changes.

The End of Software as We Know It

Late in 2025, I needed a way to process and categorize expense receipts for my company. The kind of task you would normally solve by purchasing accounting software. I spent two hours comparing options. None of them did exactly what I needed.

Then I described the problem to an AI coding agent and developed a local webapp to categorize these receipts, pre-fill the tax-relevant fields, flag anything unusual. An hour later, I had a working solution. Not a generic product bent to fit my case. A solution built for this specific moment.

After that, my *Learn and Memory Agent* improved the process. The next time I used it, the *Expenses Agent* had learned from my corrections. It pre-filled most fields correctly and generated a new interface where I only confirmed or adjusted. By the fourth time, I just said: do it. And it did.

That sequence: build me the tool, confirm what is right, learn for next time. It is not a product feature. It is a paradigm shift. But this is not even the end of the story. Why should I tell the AI agent to process the expenses at all? Why should the agent not actively remind me to submit my latest receipts, because it knows the bookkeeper has been waiting for the monthly report? At that point, the machine initiates. Not me. We are witnessing the end of software as we know it.

The Evolution of Software Interaction

To understand what is happening, it helps to trace how humans have interacted with software across its entire history. Each stage shifts responsibility from human to machine, moving from concrete artifacts to abstract context. The progression follows distinct paradigms, each with its own mental model, its own interface style, and its own assumptions about who is in control.

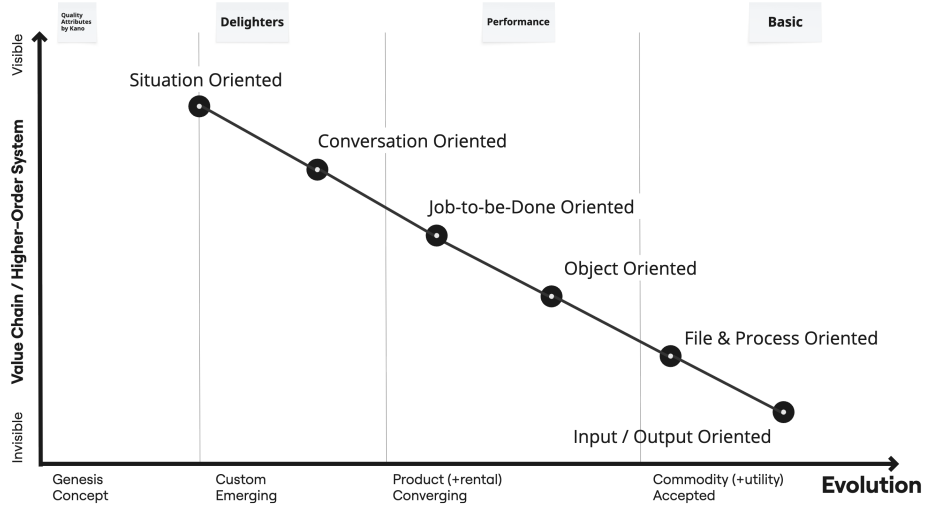


Figure 7.5.: The evolution of software interaction paradigms on a Wardley Map

Stage	Paradigm	Machine Role	Human Role	Kano Model	Evolution
Input/Output Oriented	Command-centric	Executes instructions	Operator	Basic	Commodity
File & Process Oriented	Structure-centric	Passive tool	Operator	Basic	Commodity
Object Oriented	Entity-centric	Passive tool, flexible	Operator	Performance	Product (+Rental)
Job-to-be-Done Oriented	Task-centric	Reactive to clicks	Director	Performance	Product (+Rental)

Stage	Paradigm	Machine Role	Human Role	Kano Model	Evolution
Conversation Oriented	Intent-centric	**Reactive** to language	Director	Delighter	Custom-Built
Emerging Situation Oriented	Context-centric	**Proactive**	Governor	Delighter	Genesis

Two columns deserve special attention. The Kano Model[18] captures how users perceive each paradigm. What once excited customers eventually becomes expected. File management was a revelation in the 1980s. Today nobody is impressed by "File > Save As." Conversational AI still delights in 2026, but in a few years users will simply expect it. Early paradigms have become *Basic* quality: taken for granted, noticed only when missing. Middle stages still offer *Performance* quality: the more, the better. The latest stages deliver *Delighter* quality: unexpected, differentiating, hard to compete against. A product stuck at *Basic* competes on price. One that delivers the next *Delighter* leads the market.

The Evolution column connects each paradigm to Wardley's evolution stages. Situation Oriented is at *Genesis*: a new paradigm, enabled by the components below it that have matured into *Product* or *Commodity*. The six stages that follow tell this story.

Input/Output Oriented (1950s-1970s)

The user submits commands or data. The machine returns results. No persistence, no state, no visual feedback. The user must know the exact syntax. The machine does nothing without explicit instruction.

Punch cards, command-line terminals, mainframe batch processing, early UNIX shells. The user thinks in commands and parameters. Ted Nelson's *Computer Lib*[19] (1974) was already a manifesto against this "priesthood of computing," envisioning computers as universal creative media rather than fixed command executors.

[18] see: Kano Model, 226
[19] see: Computer Lib / Dream Machines, **??**

File & Process Oriented (1970s-1990s)

The computer provides rigid structure. Users create, save, and manage files. Processes guide users through predefined steps. The mental model shifts to "I have a document" or "I follow a transaction."

IBM 3270 terminals running CICS transactions, WordPerfect, Lotus 1-2-3, SAP R/3 transaction codes. "File > Save As." Named documents on disk. Menu-driven screens. The user thinks in files and forms.

Alan Kay and Adele Goldberg's Dynabook vision[20] (1977) already challenged this paradigm, imagining every user as a programmer with "symmetric authoring and consuming." The market stayed in files and forms for two more decades.

Object Oriented (1990s, largely failed)

The user works with objects directly. A printer is an object, a document is an object. You do not "open an application." You interact with the thing itself. Objects from different sources can be embedded in one another.

OS/2 Workplace Shell, OpenDoc, OLE and ActiveX, NeXTSTEP, Taligent. Drag-and-drop objects. Compound documents. Right-click context menus on entities. The user thinks in things, not applications.

This paradigm lost commercially. But its ambition, giving users direct control over composable entities, planted seeds that only now begin to grow. An ACM survey[21] (Ko et al., 2011) found that more end-user programmers already existed than professional developers. The demand for users building their own tools was massive. The technology was not ready.

Job-to-be-Done Oriented (2007+)

"There's an app for that." One app per job. No files, no objects. I want a taxi. I want food. I want to track my run. The application disappears behind the task.

[20] see: Personal Dynamic Media, 228
[21] see: The State of the Art in End-User Software Engineering, 233

iPhone App Store, Uber, Spotify, Slack, Notion, single-purpose SaaS. App icons on a home screen. Single-purpose design. The user thinks in tasks, not tools. Onboarding asks "What do you want to do?" not "Here is your file system."

Clayton Christensen's Jobs to be Done[22] framework captures the underlying principle: customers do not buy products. They hire them to make progress in a specific situation.

Conversation Oriented (2022+)

The user describes intent in natural language. The machine delivers. For the first time, ambiguity is allowed. You do not need to know which app or which button. But the machine remains passive: it waits for the human to speak.

ChatGPT, Claude, GitHub Copilot, Microsoft Copilot in Office, and the emerging field of Malleable Software[23], where users reshape their tools at runtime through natural language.

A text input field. Natural language instead of buttons. The user thinks in wishes, not workflows.

Conversation is the last stage of the old paradigm. The human still initiates. Just more comfortably.

Situation Oriented (Genesis)

The machine initiates. It recognizes the situation, generates the appropriate interface, learns from feedback, and eventually acts autonomously. There is no pre-built application.

> "The most profound technologies are those that disappear."
> Mark Weiser, The Computer for the 21st Century[24] (1991)

[22] see: Jobs to be Done, 225
[23] see: Malleable Software, 227
[24] see: The Computer for the 21st Century, 233

OpenClaw[25], an autonomous AI agent with a Heartbeat system that initiates actions without human prompts. Claude Code, which generates tools on demand. AI agents that chain actions autonomously. Adaptive dashboards that reconfigure based on context.

No fixed UI. No app to install. The interface is generated for the moment and discarded after. The user thinks in outcomes, or does not need to think at all.

There are no applications anymore. Only just-in-time solutions.

The Software Inversion

Everything up to and including Conversation Oriented is **human-driven**. The human initiates, the machine responds. Situation Oriented is **machine-driven**. The machine acts, the human governs.

This is the fundamental inversion. Not a gradual shift, but a reversal of the basic relationship between human and machine. For the first time in the history of computing, the default mode is not "the human tells the machine what to do" but "the machine acts and the human sets boundaries."

The human role evolves across three stages:

- **Operator** (Input/Output, File & Process, Object Oriented): The human executes. They know the commands, manage the files, manipulate the objects. The machine is a passive tool.
- **Director** (Job-to-be-Done, Conversation Oriented): The human delegates. They choose the app, describe the intent, direct the conversation. The machine reacts, but only when asked.
- **Governor** (Situation Oriented): The human sets constraints and reviews outcomes. The machine anticipates, proposes, and acts. The human adjusts boundaries, or simply trusts.

The question is no longer "What should the machine do?" It becomes "What should the machine not do?"

[25] see: OpenClaw, 227

Three Steps to Autonomy

Situation Oriented computing does not arrive as a single leap. It unfolds in three sub-stages, each shifting more initiative from human to machine.

"Build me the interface." The user states intent, the AI generates a solution. Still dialog-based, still explicit. But the application is not pre-built. It is created on demand, for this specific situation, and discarded afterward. My receipt tool started here: I described what I needed, and the agent built it.

"Confirm if this is correct." The AI anticipates. It pre-fills, proposes, generates the interface before being asked. The human becomes a reviewer, not an initiator. When my agent learned my categorization patterns and presented its own suggestions for confirmation, it had reached this stage.

"Just do it." Autonomous execution. The human delegates completely. The agent processes, categorizes, files, and only surfaces exceptions. The human role reduces to governance: defining what "correct" means, setting boundaries, reviewing outcomes periodically. David Tennenhouse called this "proactive computing[26]" in 2000: the human leaves the interaction loop.

These three sub-stages coexist. Simple, well-understood tasks reach "just do it" quickly. Novel or high-stakes tasks may stay at "build me the interface" indefinitely. The appropriate level depends on the situation, not on the technology.

The business consequence is significant. Satya Nadella argues that "SaaS will collapse in the agent era[27]." Business applications are fundamentally databases with logic on top. When AI agents take over the logic layer, the traditional model of selling packaged application software loses its foundation. What remains is data and governance. Not software.

Haven't We Heard This Before?

Yes. In 1977, Alan Kay and Adele Goldberg envisioned the Dynabook: a personal computer where every user is a programmer. In 1993, Bonnie

[26] see: Proactive Computing, 229
[27] see: SaaS Will Collapse in the Agent Era, 229

Nardi documented in *A Small Matter of Programming*[28] how end users were already building their own tools with spreadsheets and macros, far beyond what software vendors intended. Object Oriented computing promised composable, user-controlled environments. Every decade has had its vision of the empowered end user.

The difference this time: the underlying components have evolved. In the 1990s, the enabling technologies for situation-oriented computing (natural language processing, on-demand compute, machine learning) were in *Genesis*. Unreliable, expensive, accessible only to specialists. Today, large language models are products approaching *Commodity*. Cloud compute is *Commodity*. The layer above them can now emerge because the layer below has matured. This is Evolution Focus in action: each evolutionary stage enables the one above it.

But the challenges are real. When every user generates their own just-in-time tools, governance becomes extraordinarily difficult. Who audits software that exists for five minutes? Who ensures compliance when the interface is generated fresh each time? For regulated environments, this is a genuine problem.

These are not objections that invalidate the paradigm. They are characteristics of *Genesis*. Every technology in *Genesis* is unstructured, hard to manage, not yet standardized. The printing press created an explosion of uncontrolled information before censorship and copyright emerged. The internet created an explosion of uncontrolled communication before regulatory frameworks caught up. Situation-oriented software will create an explosion of uncontrolled micro-applications. The governance structures will follow. But they are not here yet.

What This Means for the Enterprise

Software interaction evolves from human-operated to machine-initiated. The human role shifts from Operator through Director to Governor. And Situation Oriented computing is demanding attention.

For enterprises, the consequence is concrete: rethink your product portfolio and reorganize how you work.

[28] see: A Small Matter of Programming, 222

Your products and services sit somewhere on this evolution. Some are still File & Process Oriented. Others have reached Conversation Oriented. A few may already be experimenting with Situation Oriented capabilities. Not all will evolve at the same pace. But the direction is clear, and falling behind means competing on price in a market where competitors deliver *Delighters*.

The way your organization works must change with it. When the machine initiates and the human governs, you need less operational overhead and more governance capability. Teams shift from operating tools to governing AI agents that operate on their behalf. Leadership shifts from directing specific solutions to defining the constraints within which AI-generated solutions must operate. The AI-enhanced team[29] is the organizational expression of this inversion.

Map where each of your products sits on the evolution path. Identify the next step for each. Reorganize to match. This is Evolution Focus in practice (see Developing a Strategy for the GenAI Era).

The end of software as we know it is not the end of the need for structure. It is the beginning of a new kind of structure: one designed for governing intelligence rather than operating tools.

[29] see: The AI-Enhanced Team of the Future, 117

8. Methods

The Good, the Bad, and the Ugly about (Agile) Frameworks

A Guide on how to Choose and use Frameworks for Your Organization Wisely

Frameworks for organizations are prevalent nowadays. Scrum[1], SAFe[2], LeSS[3], Cynefin[4] claim to be frameworks. Kanban[5], Flight Level Kanban[6], Lean[7] and ScALeD[8] are often named frameworks, even though their authors don't claim it. And there are even more on the market.

A framework typically promises to change your organization to improve specific aspects. A framework associated with *Agile*[9] indicates that the organization becomes more agile with its application. In other words, your company can react more effectively to changes in the market or any other complex domain. Please note, some of the mentioned Frameworks have other objectives.

This also applies to AME3, which is designed with the primary objective of advancing the evolution of the enterprise. It is even referred to as a Meta-Framework.

The Good

You don't Need to Reinvent the Wheel

Other organizations are successful with the framework, so should we. We can easily upload experiences and knowledge by learning and using a framework. That's why most frameworks come with an education plan and material. So we can skip all the hassle with trial and error. Nice!

[1] see: Scrum, 210
[2] see: SAFe, 207
[3] see: LeSS, 201
[4] see: Cynefin, 189
[5] see: Kanban, 199
[6] see: Flight Levels, 196
[7] see: Lean, 201
[8] see: ScALeD, 207
[9] see: Agile, 185

Strategy for Free

The market and the environment are changing ever faster. So companies, their teams, and employees need to change faster to keep up with their services and products. But in which direction? Frameworks promise significant changes for organizations in a short time and offer a build in strategy ready to adopt. Therefore, using a framework is a competitive advantage. Not using it, when competitors are successful with it, is a risk. It would be like running your own server-farm while your competitors are using cloud-computing.

New Culture

> Culture: the ideas, customs, and social behavior of a particular people or society.

The culture of an organization becomes visible through the behavior of its employees or, better, through their habits. Every organization grows its culture, and that is a good thing. Scaling and growing into a market need standards followed by the employees without thinking too much.

However, we often experience the need to change such organizational habits as soon as we want or need to change our products and services. Frameworks bring new behaviors into an organization so that employees learn a new culture.

Scrum, for example, has a clear description of behavior, and people are told: just follow the book. You will understand later. So the Agile Mindset (a.k.a. Agile culture) comes through the back door. Scrum even has a measurement to stop old behavior and helps to unlearn things: The ScrumMaster. Craig Larman summarized this in his Law: *Culture follows structure*[10].

[10] see: Larman's Laws, 200

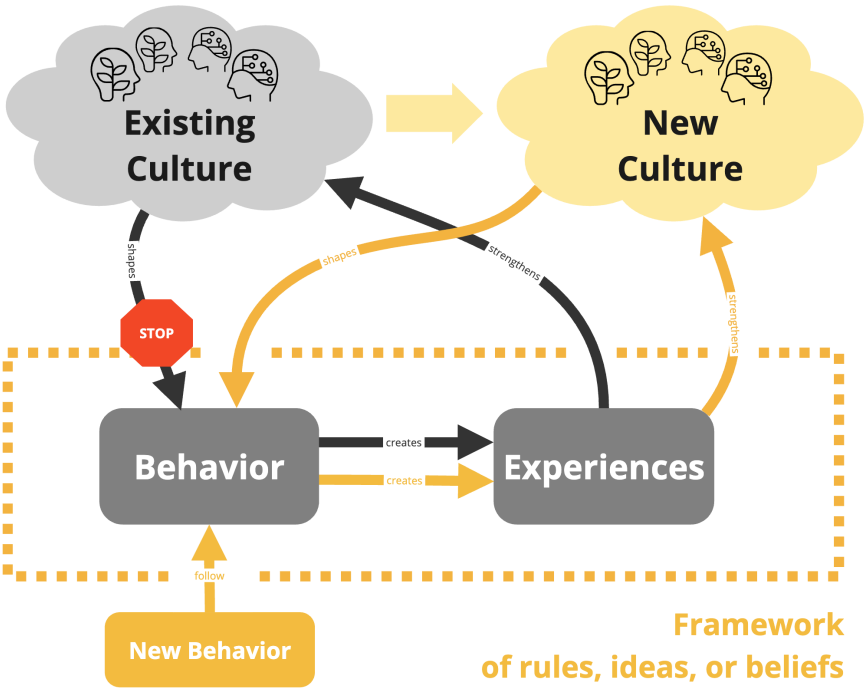


Figure 8.1.: Culture follows Structure — How frameworks influence the system dynamic

The Bad

Promising Simplicity where there is None

> Culture eats strategy for breakfast – Peter Drucker

Have you ever looked at a graphic showing Scrum or any other framework and thought: That looks simple? Well, it is simple in terms of the rules, structure, and elements (not all frameworks, though). But every practitioner knows it can take years to adopt it, even with particularly simple frameworks like Scrum.

As mentioned before, a framework is changing the culture. That means changing habits. And it needs a lot of repeating and fallbacks to change

them. Just imagine trying to stop smoking. It is actually a straightforward thing. Just stop it. You even know that it harms your health, but you believe lung cancer only affects others. Cutting off old behavior and following new is pretty difficult. For example, by putting chewing gum in your mouth instead of lighting a new cigarette. Especially in a stressed situation, your subconscious will choose the cigarette. Now, imagine that for a team or even for a whole organization.

It doesn't Solve Anything yet

> Framework:
> (1) a supporting structure around which something can be built
> (2) A set of assumptions, concepts, values, and practices that constitutes a way of viewing reality
> (3) A system of rules, ideas, or beliefs that is used to plan or decide something
> (4) The ideas, information, and principles that form the structure of an organization or plan

A framework is just a basic structure and not a keyhole ready house you can live in. The same is for organizational frameworks. Some cornerstones, beams etc. are fixed. But you have to fill large spaces in between to run your business. A software team using Scrum needs knowledge, practices, and methods to develop software. But Software is not even mentioned once in the Scrum description, even if Scrum became popular in software development.

That has its benefits as described at the beginning. Yet, it leaves organizations with a lot of uncertainty. Employees are asking for more details. They want to be "SAFe" to do it right. The problem, though is, that the uncertainty comes from the nature of your business, not from the framework or methods you use. Frameworks can disguise this fact.

You May End Up with Something You Probably Don't Want.

A framework follows the idea that you can repeat what has been observed at one or many organizations. We believe by applying the framework, we get a certain effect within our organization. But was the success of the observed organizations because of the framework, or because they had some brilliant

employees with the right product idea at the right time? Correlation is a bitch. Furthermore, the believed effect why a framework is used often vary or is hidden. For example, I have observed a driving, but hidden belief to apply SAFe in many organizations: Increase the amount of better paid management positions and consulting contracts.

The Ugly (or the beautiful) way to Choose and Use Frameworks

How to Choose

1. **Make first the goal clear, then choose the framework.** For example, if you find out, that you are operating in a complex domain and you want to increase the effectiveness of your work system (not efficiency), Scrum or LeSS is maybe the right choice. If you would like to increase the yield in your production, you should consider Lean or Kanban. Before you define your goal, you should first analyze the status quo of your product, services, and organization and then derive your strategic goal. The Cynefin framework or Wardley Mapping offering help here.
2. **Trust the long track record of frameworks, but don't rely on them entirely.** I fear many testimonials are written to please the authors, the organization or the frameworks themselves. And since we are dealing with complex social systems, correlation does not equal proof. Case studies creating a false sense of security. Consider every use of a framework or method as an opportunity for experimentation and learning.
3. A good property for frameworks should therefore be a **built-in inspection and adaptation** measurement. This ensures that the spaces between the frames are filled and exchanged with additional practices and methods over time. Or that the abandonment or dissolution of the framework is recognized at an early stage. This inspection and adaptation mechanism should always be toward the goal, why we are using the framework in the first place. For example, the LeSS framework includes the Overall Retrospective and the community of ScrumMaster for this purpose. If a framework lacks elements like this, it should definitely be added. For example, they can be integrated into the Tournament or Match of the AME3 framework.
4. **Find the right balance between fixed structure and freedom to experiment.** Some frameworks just define some principles and

leave huge space for the organization to fill them. Many experiments need to happen to fill the spaces. This puts very high demands on the ability of employees to adapt. On the other hand, while a descriptive framework provides more stability, it may lack the flexibility to cater to individual needs. Companies then often circle around the framework and try to adapt to its structure and rules, thereby losing sight of their own business objectives. Perhaps Scrum rose to popularity because it struck the sweet spot of descriptiveness, which many technology-driven organizations require. In case of doubt, choose the simpler frame.

How to Use

1. **A good framework uncovers your organization's weak spots.** It's up to you and your colleagues to leverage this insight for improvement. For example. If the Cynefin framework reveals that the essence of your product domain is complex, yet you persist in applying measures suited for the simpler domain, the responsibility to adapt rests with you.
2. **It's crucial to secure commitment in your organization** to making the necessary adjustments as you begin working within the framework's structure. Clear framework guidelines are essential in this respect. I often encounter skepticism when I mention that Scrum explicitly allows only three roles within a Scrum Team, despite individuals claiming years of Scrum experience. Ignoring this creates waste and makes using the framework pointless. It is then even better not to use the framework.
3. **Divide and conquer.** Begin by implementing a framework in smaller parts of your organization, ideally in areas where it can be applied without disrupting the rest of the company. This approach is essentially the only way to develop expertise effectively. It reduces the risk of undesired development. If the framework proves beneficial, you can then extend this expertise to other parts of the organization. Once your colleagues become experts, you may craft your **own framework** or organizational design upon the initial one.

Spotify[11] exemplifies this strategy. They started with Scrum with a few teams and, leveraging that expertise, developed their own structure upon

[11] see: Spotify, 212

Scrum to scale their product and organization. However, this structure was tailored specifically for Spotify, continuously evolving and may not be directly applicable to other organizations, unless they are in the business of developing and maintaining a music streaming service.

The Meta-Framework AME3

AME3 takes a unique approach. Its main goal is to drive the evolution of the entire company, its products and services. To achieve this, existing methods and frameworks are used by incorporating the previously mentioned recommendations. The "M" in AME3 stands therefore for "Meta-Framework" and is a testament to this approach. And of course there must be a reason for the "M" in the acronym.

Conclusion

Navigating the landscape of organizational frameworks requires a thoughtful approach, blending analytical rigor with practical experimentation. The allure of frameworks like Scrum, SAFe, LeSS, and other lies in their promise to usher in agility and a new culture within organizations.

But Frameworks can achieve a lot more than agility. In a good and a bad way. Their successful implementation demands more than just blind adherence to principles or an uncritical adoption driven by trends. It requires a clear understanding of your organization's goals, a willingness to adapt and iterate based on real-world feedback, and, perhaps most crucially, a commitment to fostering a culture that values continuous improvement over rigid conformity.

By choosing and using frameworks wisely, organizations can harness their potential to drive meaningful change, unlock new levels of performance, and thrive in an even more complex and faster pacing world.

Version: 2

The Project to Product with Scrum Playbook

An AME3 Decision-Making Guide for Project-Centric Organizations

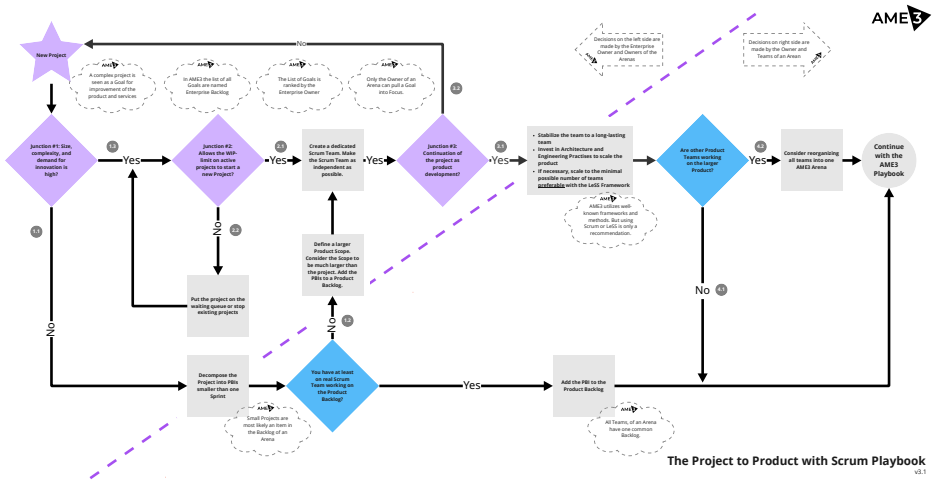
The demands for agility in projects are increasing more and more because the surrounding conditions such as customer requirements, technological progress and society are changing at ever shorter intervals. Scrum has therefore become an essential part of everyday project work in companies. It is not unusual for a project to require several teams and Large Scale Scrum (LeSS) is the consequent choice here. However, project-centric organizations often waste potential and resources when they try to combine projects with long-term product and service development.

The guideline presented here helps to make decisions at the right time whether and how Scrum or even LeSS should be used for projects in the organization. In doing so, the advantages resulting from the product mindset of Scrum can be better utilized.

By following the playbook, the first steps are made to move the project portfolio forward to a decision-making instrument for investing in innovation and improving the enterprise's value creation. We simply call that Enterprise Backlog. Furthermore, an Arena will continue in advancing the Product. This is a starting point for an evolution-based organization design based on AME3.

Alternatively, money has been saved because the project was stopped as soon as the early results indicated that the promised ROI would not be achieved.

All junctions of the playbook are shown in the following image. Detailed description of each junction and routes can be found in the following article by its number.

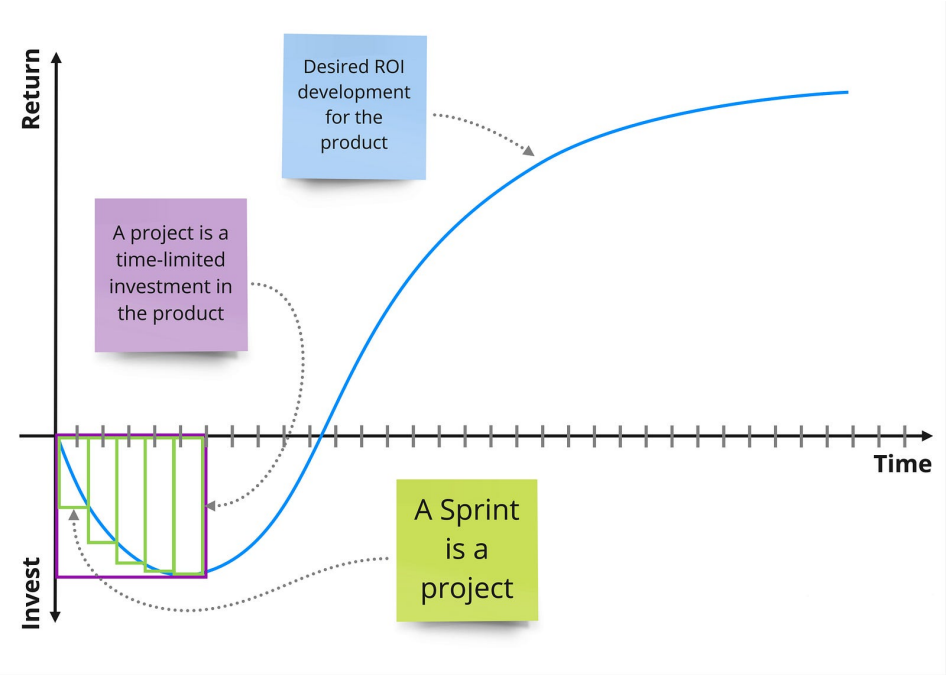


Project vs. Product

> „A project is a temporary endeavor undertaken to create a unique product, service, or result.“

This is the definition of a project according to the Project Management Institute (PMI). A project therefore has a fixed time frame in which investments are made to achieve a result, e.g. an improvement in the production process or a software system. This is a significant difference to a product. The lifetime of a product is not fixed. In fact, we as a company are constantly investing in the product. Primarily with the work of our employees. Even if we want to get rid of the product once, we have to invest. It is ultimately the last user of the product who determines when a product no longer exists. Perhaps the employee of a waste disposal company.

Products are therefore significantly larger in scope than projects and the targeted performance is anything but linear and also difficult to predict.



The blue curve in the image should be considered as the desired development of the ROI. From a company's perspective, a project is an investment in the product for a certain period of time (red box in the image). Scrum is a framework that is helping to optimize the value generated for this investment. Scrum can be understood as a sequence of projects with a fixed period of one sprint (green boxes in the image). Scrum thus generates knowledge about how the next investment in the product can be improved at often much shorter intervals as before. In extreme cases, this knowledge can stop the investment in the product completely at an early stage. So, how should you handle a new project if you want to think in terms of product development right from the start?

Junction #1: Simple, Complicated or Complex?

(1.1) Short Project Duration, Complicated Undertaking.

In many companies, there are projects that are smaller than a Sprint[12]. This means that they are completed within 4 weeks or less. For example,

[12] see: Sprint, 213

the customer projects of a web agency. Setting up a separate Scrum team for each of these projects, or rather customer orders, would take too much time and effort to build up the team. Instead, it is advisable to include the customer order as a Product Backlog item and a Scrum team pulls the item in one of the next sprints. The team can then optimize its work with the customer so that waiting times are minimized and a customer order can be completed within a sprint. This increases value and flexibility for both the customer and the web agency.

Some of the web agency's client assignments may stretch over 2–3 sprints. Nevertheless, the amount of work is relatively small and well understood. However, there are always waiting periods because decisions have not yet been made by the customer, the customer's work packages have not yet been completed or things are only to be delivered on a certain date. The picture below shows the customer projects of a software company on a Kanban system. The backlog entries for training by the support team, customization of the standard software by the software team and provisioning of the infrastructure were created by the team members and ranked accordingly in the Backlogs.

(1.2) At this stage, you may not have a real Scrum team in the company at all. To start with, you should first ask, what is the product that the employees are working on? To find an answer, it helps to look at the meaning of the term "product":

> "The product is the result of work".

In the case of the web agency, the product could therefore be: "The Service for the development of web offerings". The scope or limits of the product must therefore be set particularly broadly. If so, a Scrum team can be formed to improve and deliver the service. It is then also clear that in the case of the web agency, the Scrum Product Owner role is provided by the web agency itself and not by the customers. This is because the agency owns the service and bears the associated obligations. An essential task of the Product Owner role is to evaluate the customer orders according to an internal customer ranking and to position them in the Product Backlog. Even if each item is only complicated, operating and improving this service as a whole can be very complex.

There are projects whose context is very well understood and are constantly in the complicated or simple domain. For example, the customer projects of a prefabricated house manufacturer. Here, they rely on serial production to reduce unit costs and to scale in the market. The fewer variants you allow in the houses, as less complicated the work system will be. And the greater is the potential for mass scaling, of course. The team or teams primarily consider the result of the production and construction as their product. In other words, the turnkey houses. They derive the necessary optimizations from the experience gained from the customer projects and take these into account in their Product Backlog. To do this, planners, manufacturers, fitters and craftsmen in the teams break down these improvements into suitable Product Backlog items together.

For example, one team can advance the further development of the smart home control system, while another team deals with the automation of series production and a third team takes care of the semi-automated order and planning process. I recommend using Wardley Maps by Simon Wardley to identify multi team structure and potential for innovation.

> [!info] In AME3 small projects are most likely one or a set of Improvements in the Arena Backlog

(1.3) Complexity and Need for Innovation is High

The majority of projects in companies usually have a significantly longer duration than 2–3 sprints. A longer duration is an indicator of higher

complexity. Complex means that many unknown factors will influence the project that cannot be predicted at the start of the project. This is not a failure of the project team, but is simply the nature of projects and product development. One factor is the constantly changing requirements. Another is technical realization because many solutions are uncharted territory.

Very innovative projects are characterized by the fact that we only work with very vague assumptions for both factors. We can't even say how many changes we will have and in what quality. It is in the nature of things that it is difficult to determine the degree of complexity. As a simple guiding principle, we can state: If the question whether the project is complicated or complex needs to be discussed. Then it is most likely complex. However, David Snowden's Cynefin Framework can provide guidance here.

[!Info] In AME3 a complex project is seen as a Goal

Junction #2: Does the WIP Limit for Active Projects Allow You to Start a New Project?

So far, we have brought the project to the team or teams. The advantage is obvious. We don't have to invest in setting up a new team. The employees are also grateful because they don't have to constantly reorganize themselves and can focus on results. The work more likely in a flow state, and we are talking about real, effective teams.

However, if our project is complex and our demand for innovation is high, we come to the standard case for Scrum. This is actually the birth of every Scrum environment, as described in The New New Product Development Game[13] by Ikujiro Nonaka and Hirotaka Takeuchi. We look for the right talents to form an initial, new team. A Scrum Master or System Lead takes care of building the team. A manager with decision-making powers takes on the role of Scrum Product Owner or Owner. This team, which is as autonomous as possible, can now question existing rules and develop entirely new approaches to solving the problem. To a certain extent, we can see such a team as a start-up within the company. Some companies even go down the path of funding a real start-up.

(2.1) But beware, many companies fall into a trap here. Too many projects are started at the same time. There are companies whose number of active

[13]see: The New New Product Development Game, 214

projects is greater than the number of employees. A complex project does not just affect one team. Often several teams are involved and the work of the entire organization is influenced. The project team or teams are therefore anything but autonomous. It is difficult to predict how many parallel projects a company can carry out without causing a traffic jam or a complete blockage in the work system. However, it is safe to say: fewer than one might first assume. The fewer the number of active projects, the higher the lead time. This generates value and feedback earlier, which in turn benefits the next projects.

(2.2) So before starting the next project team—with or without Scrum—it is essential to check that the maximum reasonable number of active projects is not exceeded. If this is true, the project must be weighted against all other projects. Are those organized with Scrum, then the Increment[14] and the experience of the team(s) is providing the best possible data for the decision makers. A positive result of this step can therefore be, to cancel an active project in favour of a new, more attractive project. It makes no sense to throw good money after a bad investment.

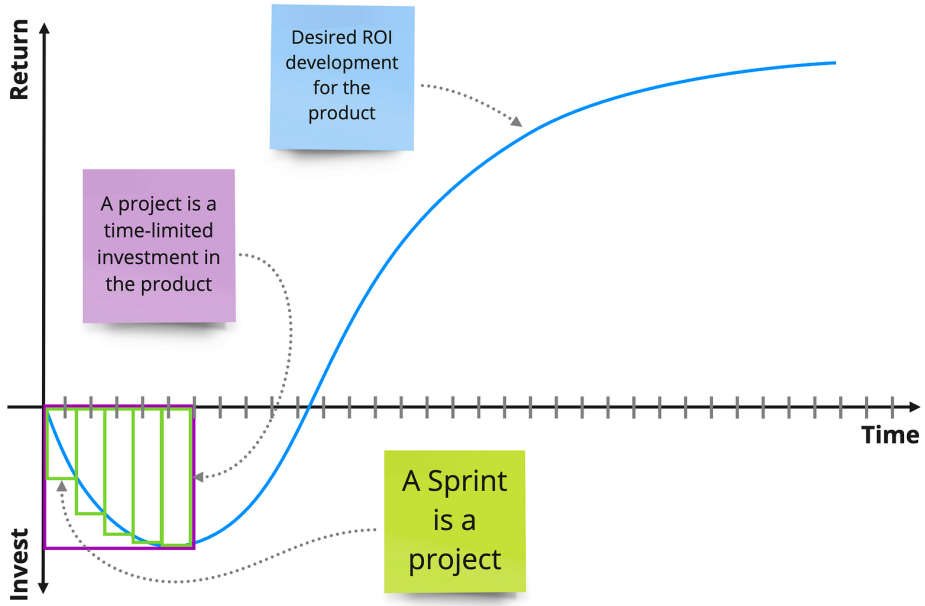
Finally, a project can be considered as an entry in the company's project backlog. In AME3 simply named Enterprise Backlog. Many companies often no longer use the term project at this level. Personally, I prefer the term innovation Goal. Because that's what we want to do. We intend to innovate with our product and set ourselves a goal. In any case, a project profile that outlines the most important parameters is sufficient. With the Goals or project profiles as entries, the Enterprise Backlog can be the input for a Kanban system to make the flow from concept to cash transparent. With such a system, you can now explore how high the maximum parallel project load can be, by experimenting with a work-in-progress limit (WIP-limit) for Goals or projects.

The picture below shows such a Kanban system after an initial workshop. It shows all known projects in the pipeline. Some projects have a volume of more than 10 million € and an estimated project duration of more than 2 years. However, the option column is not yet an Enterprise Backlog. This is because most projects of the Enterprise are missing. It only displays the projects that the potential new Arena will likely be influenced by in the future. Well, at least the workshop participants quickly agreed that the pipeline is already jammed.

[14] see: Increment, 199

[!info] In AME3 the list of all Goals (projects) are named Enterprise Backlog, prioritized by the Enterprise Owner. Only the Owner of an Arena can pull a Goal into focus.

Junction #3: Continue the Project as Product Development

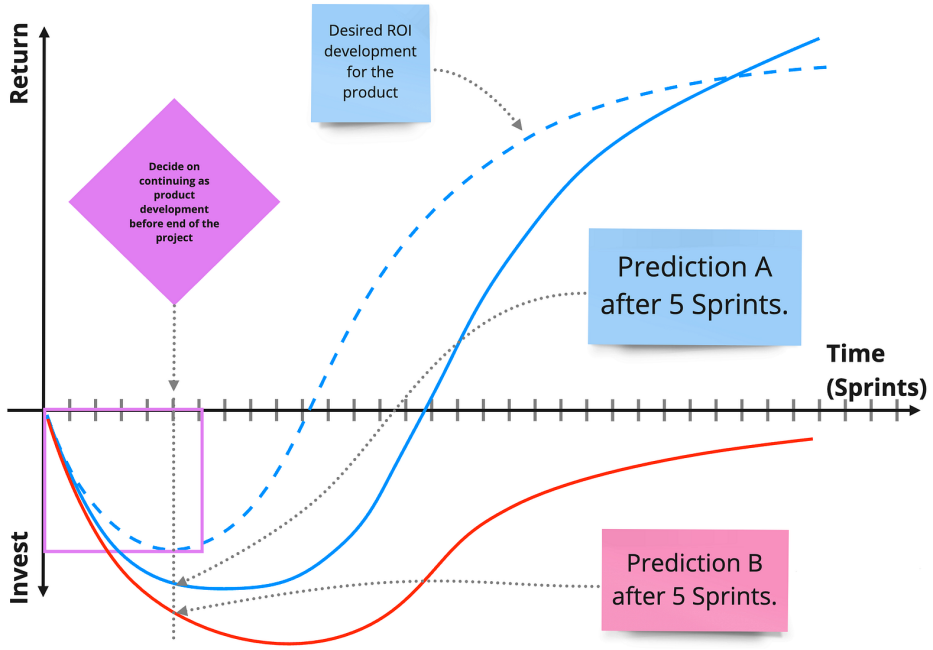


(3.1) With the description *Desired ROI development of the product* for the blue curve in the picture, you can already guess. The reality will probably be different. Whatever the innovation project or Goal was, the costs are most likely higher than expected. Scrum is now showing its full strength because in accordance with the principles of the agile manifesto[15], progress is measured by the finished product Increment. It is the best source of data for making predictions about the future. You get more information with every sprint. Usually after 3 sprints, at the latest after 9 sprints, a decision can be made whether to continue with the product development. It is crucial to consider now the following:

1. Stabilize the team and do not expand it unnecessarily. Far too often, a new employee is hired for every new problem. It is more important to keep the social complexity for the team members to a minimum. Perhaps external staff have been brought in and you should now secure the talent on a permanent basis. The project team may become a new organizational unit.
2. The product owner should pay particular attention to the non-functional requirements or aspects of the product and focus the team on technical excellence and a solid system architecture. More important than scaling employees in the working system is being able to scale the product in the market. For example, should the teams be able to prove through experiments how they intend to ensure the exponential user growth the product is aiming for? Or what architecture is needed to be able to constantly deliver new features?
3. If the product team grows to more than 10–12 people, the Scrum-Master should choose a lean framework for scaling. If done skillfully, scaling according to LeSS will happen by itself. For example, by following the principle "LeSS is multi team Scrum".

(3.2) As previously mentioned. A win can be the early conclusion that the new product and thus the project will not be a success as expected. Now it becomes clear how essential it is to have a Product Owner who actually takes ownership. She must ensure that the strategy on the flight level of the project and innovation Goals is changing. This clears the way for a project that can open up more opportunities for the company.

[15]see: Manifesto for Agile Software Development, 203



In case A, you will probably continue to invest in the product. In case B, you will most likely cancel the project and product development.

> [!info] In AME3, the Owner and Enterprise Owner are responsible for deciding whether to continue working on a project (Goal) or to change direction by removing the current Goal and focusing on a new one.

Outlook: Switching to an AME3 Arena

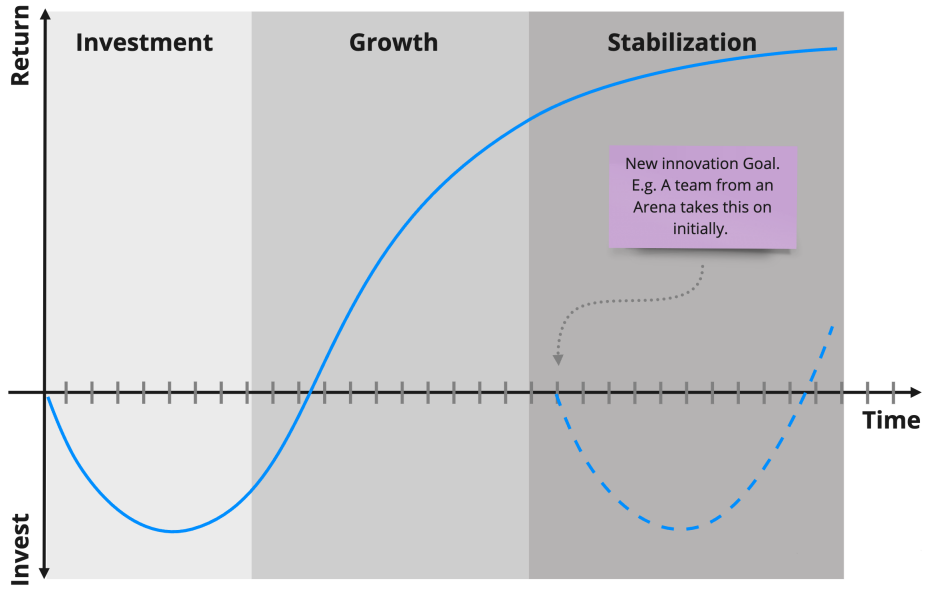
> [!info] Following the playbook here, an Arena in the sense of AME3 has already been established.

Now that the project has produced a promising result and a new, Scrum-based working system has been piloted, the question arises as to if and how we can use this for the entire organization.

(4.1) It may be possible for our newly created project organization to work very independently of all other parts of the organization. There is much to be said for leaving the organizational unit as such. We call that unit Arena in AME3.

> [!info] AME3 leverages established frameworks and methods. However, employing Scrum or LeSS is merely a suggestion. For example, an Arena may include organizational components within the complicated domain that are not suited for optimization through Scrum or LeSS.

(4.2) If other teams are also working on the same or related products, it makes more sense to use the new expertise and incorporate it into an overall product organization, a.k.a Arena. Of course, this means a reorganization of the teams and the organizational units involved. In the end, the subproduct created initially by the project is further developed by all teams of an Arena. Such an organization then no longer speaks of projects but, as mentioned above, of Goals.



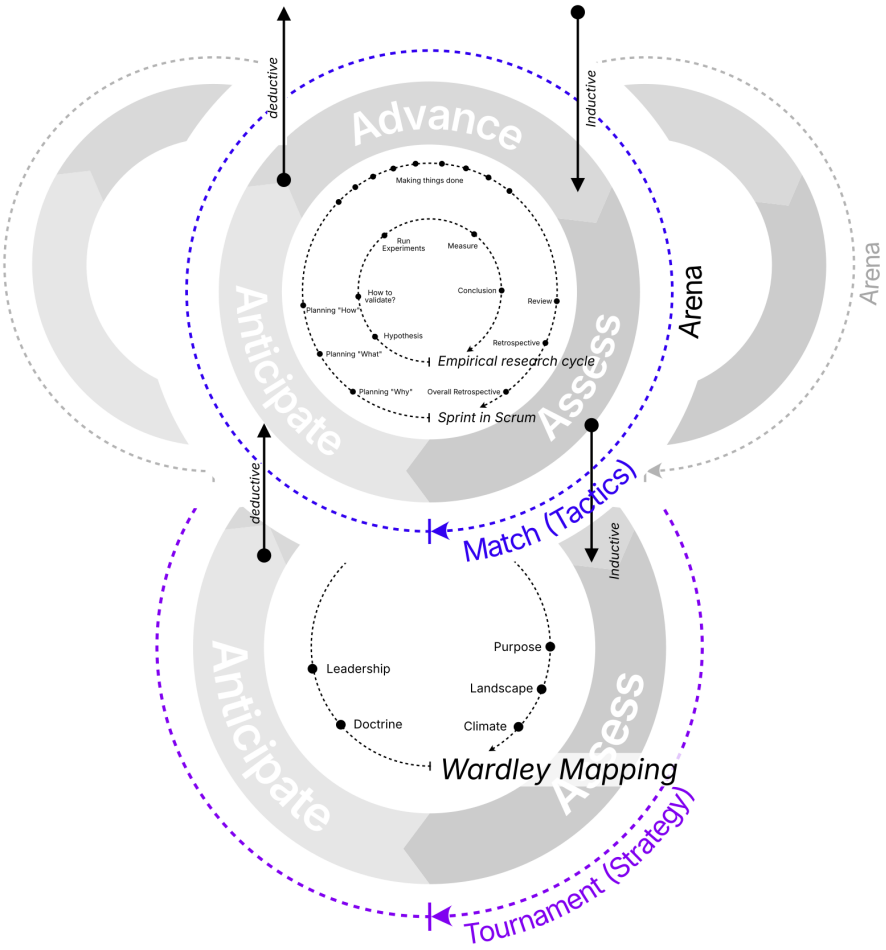
This is particularly advantageous when the further development of the Product enters the stabilization phase. In other words, we only have an interest in ensuring that customers can continue to generate value, but we do not expect innovation and therefore massive value growth. The Accountable Representative of an Enterprise should now refine the strategy toward the next step in the evolution of the product and services. The result will be

new Goals for innovation. An Arena can pull that Goal and a Team start working on it using Scrum.

Example Mapping of Methods and Frameworks

The Tournament and the Match are designed to support well-established methods and frameworks. The following illustration demonstrates how Wardley Mapping, Scrum, and a typical empirical research cycle can support each other in an AME3 organization.

Many other combinations are possible and used in practice. However, which methods and frameworks to combine depends on the situation, Ambition, and Goals of the Enterprise.



The Estuarine Framework

The Estuarine Framework is the third major framework in the Cynefin ecosystem, developed by Dave Snowden. It provides a model for understanding the forces that shape an organization — constraints, constructors, and actors — and for identifying where small interventions can shift the system.

Estuarine Mapping is the facilitation method used to apply the frame-

work. In a structured workshop, participants identify the forces at play, place them on an energy/time grid, and design interventions. The map is the output. The framework is the thinking behind it.

The Estuarine Framework complements AME3's Empirical Control and Evolution Focus doctrines. Where Wardley Mapping reveals the competitive landscape and direction of evolution, the Estuarine Framework reveals the internal and external forces that enable or resist change. Combined, they give an Enterprise both a strategic compass and a constraint map.

The Estuary Metaphor

An estuary is where river meets sea. The water flows in and flows out. Some elements are stable like a granite cliff, checked rarely. Others, like sandbanks, shift daily. As the tide turns, some elements become visible or disappear entirely.

This metaphor captures how organizations actually work: multiple flows of possibility, not a single linear path. Some things you can only change at the right moment. Some forces are visible, others hidden. And the landscape shifts constantly.

Four Types of Estuaries — Four Types of Organizations

Geologists classify estuaries into four types based on how they formed. Each type maps to a recognizable organizational pattern:

Estuary Type	Formation	Organizational Analogue
Coastal Plain (Drowned River Valley)	Rising sea levels flood existing river valleys. Wide, shallow, open to the sea. Examples: Chesapeake Bay, Delaware Bay	**Legacy enterprises absorbing market change.** The existing structure (the river valley) was built for a different era. Now external forces (rising waters) have flooded it. The old channels still shape the flow, but the landscape has fundamentally changed. Most large enterprises undergoing digital or GenAI transformation are coastal plain organizations.

Estuary Type	Formation	Organizational Analogue
Bar-Built (Restricted Mouth)	Ocean waves build sandbars or barrier islands, creating sheltered lagoons behind them. Water exchange with the ocean is limited. Examples: Pamlico Sound, Matagorda Bay	**Protected organizations with restricted market exposure.** Internal IT departments, regulated monopolies, or government agencies. A barrier (regulation, captive customers, funding model) shields them from the full force of market dynamics. Calm inside, but also stagnant. Heavy rains (crises, political pressure) can break through the barrier and force sudden change.
Tectonic	Crustal movement causes land to sink, creating new basins that fill with seawater. Examples: San Francisco Bay	**Organizations disrupted by structural shifts.** A tectonic event — a merger, a market collapse, a new technology like GenAI — creates an entirely new basin. The old terrain has sunk. Fresh water (internal capability) and seawater (external demands) mix in a space that did not exist before. Startups in new markets and post-merger organizations are tectonic estuaries.
Fjord	Glaciers carve deep, narrow channels. When the glacier retreats, seawater fills the valley. Deep water, narrow mouth, limited exchange. Examples: Puget Sound, Glacier Bay	**Deep-expertise organizations with narrow interfaces.** Research institutions, specialized engineering firms, or niche consultancies. Deep knowledge (deep water), steep walls of expertise, but a narrow sill at the mouth limits how much outside influence gets in. The bottom can become stagnant and anoxic — brilliant capability that never reaches the market.

Understanding which type of estuary your organization resembles helps you choose where to focus your mapping. A coastal plain organization has different constraints than a fjord. A bar-built organization will respond to different interventions than a tectonic one.

Core Concepts

The Estuarine Framework works with three types of **actants** — the forces that shape a system.

Constraints

Constraints shape systems and affect patterns. They are not merely barriers. A constraint can enable, govern, connect, or contain.

Type	Description
Governing	Laws, rules, codes — provide stability but are sensitive to change
Enabling	Heuristics and principles — allow distributed decision-making
Connecting	Flexible, adaptive structures — links across boundaries
Containing	Clear boundaries — departments, categories, defined scope
Rigid	Resist change until breaking point — deadlines, compliance requirements
Dark	Effect visible, cause unknown — hidden cultural forces

Constructors

Constructors produce consistent, replicable transformations. They change things that pass through them. A hiring process, an onboarding program, or a deployment pipeline are constructors. They transform through:

- **Passage** — things change by moving through the constructor
- **Contagion** — things change by proximity to the constructor
- **Presence** — things change simply because the constructor exists

Actors

Actors are individuals, roles, collectives, or entities with intelligence and intention. They are the people and groups who act within and upon the system.

The Energy/Time Grid

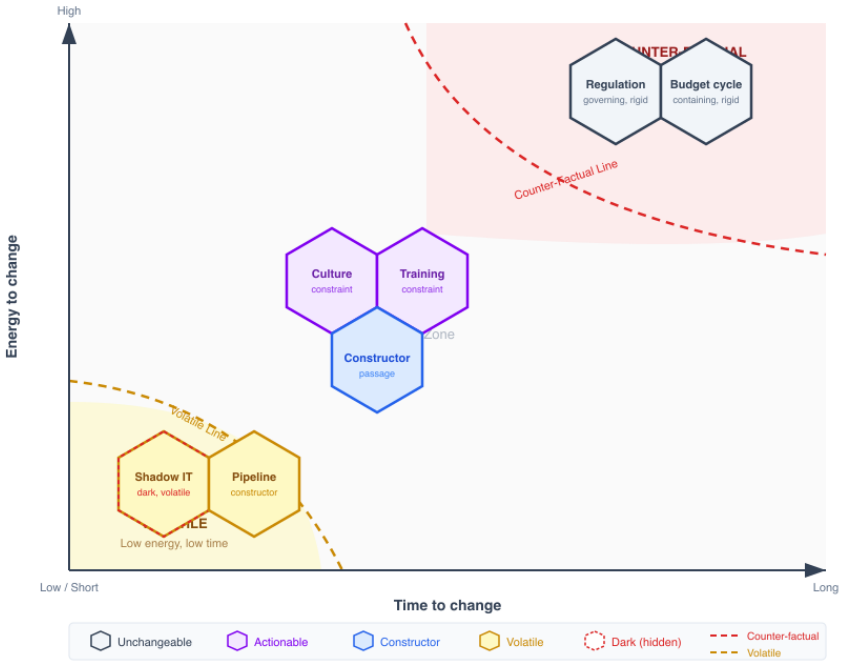


Figure 8.2.: The Energy/Time Grid with the counter-factual and volatile zones

During an Estuarine Mapping workshop, all actants are placed on a two-axis grid:

- **Energy** (vertical): How much effort, resources, or political capital is required to change this actant?
- **Time** (horizontal): How long would it take to change it?

This creates a landscape where the position of each actant tells you something about where change is feasible and where it is not.

The Two Lines

Two lines are drawn on the grid to separate distinct zones:

The Counter-Factual Line separates the top-right corner from the rest. Everything above this line is, for practical purposes, currently unchangeable. These are the granite cliffs. You work around them, not against them.

The Volatile Line isolates the bottom-left corner — actants that require very little energy and time to change. These are the sandbanks. They can shift quickly, which means both opportunity and risk. Low cost of change does not mean low impact.

Estuarine Mapping: The Seven Steps

The mapping process follows a pre-process step and seven numbered steps. Steps 0–4 build the map. Steps 5–7 turn it into action.

Step	Activity
0	**Pre-process**: Define the question. Collect material — anecdotes, observations, data
1	**Identify actants**: Brainstorm constraints, constructors, and actors. Explore hidden ("dark") actants
2	**Map on energy/time grid**: Place actants, cluster similar items
3	**Draw the counter-factual line**: Separate what cannot practically be changed
4	**Draw the volatile line**: Identify low-energy, low-time actants and assess their impact
5	**Design interventions**: Generate safe-to-fail experiments using the action categories
6	**Set direction of travel**: Establish monitoring systems and review cadence
7	**Combine into portfolios**: Aggregate micro-interventions into a coherent strategy

Action Categories

Interventions in Step 5 fall into three categories:

Vector Actions — change the position of actants on the grid:

- *Compass Rose*: Shift energy or time costs up or down
- *Destroy*: Eliminate an actant

- *Stabilize*: Keep an actant where it is

Signal Actions — sense and respond:

- *Conditional*: Identify links between actants
- *Monitor*: Observe, especially near boundary lines
- *Trigger*: Directly change an actant when conditions are met

Communication Actions — create new connections:

- *Research*: Investigate, sense-make, extend options
- *Request*: Seek collaboration, outreach, or permission
- *Interaction*: Change the connections between actants

In Practice: Using GenAI in a Public Sector IT Organization

Katrin leads an internal IT department of 800 people in a German state ministry. Her teams maintain the tax processing systems, citizen portals, and internal infrastructure. For months, she has watched commercial GenAI tools spread through the ministry — department heads using ChatGPT for drafting policy documents, developers experimenting with Copilot on their personal accounts, citizen service teams asking when they will get an AI assistant.

None of this is sanctioned. None of it is governed. And none of it will stop on its own.

Katrin decides that ignoring GenAI is no longer an option, but she also knows that launching a "GenAI Strategy Program" would take eighteen months just to get through procurement. She needs a different approach — one that works with the forces already in motion rather than trying to control them from above. She commissions an Estuarine Mapping workshop.

Recognising the Estuary

The first insight comes before anyone touches a sticky note. The facilitator asks the group: "What kind of estuary are you?"

The answer is obvious. This is a **bar-built estuary**. Regulation, procurement rules, and the staff council form a barrier that shields the organization

from direct market forces. Inside the lagoon, the water is calm. Exchange with the outside is limited. The challenge is not surviving the ocean — it is preventing stagnation.

This framing shifts the conversation. The question is no longer "How do we catch up with the private sector?" It becomes: "What is already flowing inside our lagoon, and where are the openings?"

Surfacing the Actants

Over two hours, the group identifies the forces shaping their system.

Constraints:

- *Data protection regulation (DSGVO/GDPR)* — governing, rigid
- *BSI cloud security requirements* — governing
- *Procurement rules (Vergaberecht)* — governing, rigid
- *"We've always done it this way" culture* — dark constraint
- *Staff council co-determination (Personalrat)* — governing, but also enabling
- *Annual budget cycle* — containing, rigid
- *Shadow IT with commercial GenAI tools* — dark constraint, already happening, uncontrolled
- *Cross-department data silos* — containing

Constructors:

- *Existing DevOps pipeline* — transforms through passage
- *Internal training academy* — transforms through passage
- *Innovation lab (small, underfunded)* — transforms through presence
- *Monthly IT leadership round* — transforms through presence

Actors:

- Katrin (CIO/sponsor), IT architects, department heads, staff council, data protection officer, vendor partners, citizen service teams

The dark constraints provoke the most discussion. Shadow IT is not on anyone's risk register, but everyone in the room knows it is happening. The "we've always done it this way" culture is harder to name — it shows up as slow approvals, risk aversion in architecture decisions, and a preference for studying problems over solving them.

Building the Map

The actants go onto the energy/time grid:

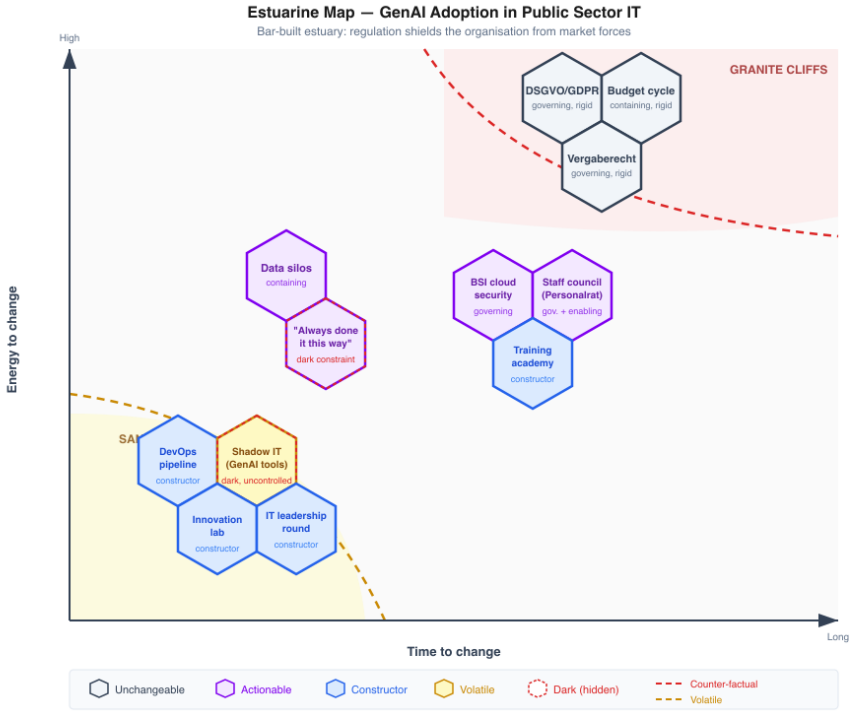


Figure 8.3.: Estuarine Map — GenAI adoption in a German state ministry IT department

Above the counter-factual line: DSGVO, Vergaberecht, and the annual budget cycle. These cannot be changed by this group. They are granite cliffs. All GenAI initiatives must work within these boundaries. Katrin makes this explicit: "We stop talking about changing these. We design around them."

Inside the volatile zone: The DevOps pipeline, the innovation lab, shadow IT usage, and the IT leadership round. These can change quickly with low effort — but their current state is unstable. Shadow IT in particular is a sandbank: it shifts daily and carries risk. One data breach and the staff council will shut everything down.

Designing Interventions

With the map in front of them, the group moves from analysis to action. Each intervention is small, reversible, and designed to shift specific actants.

Actant	Action	Intervention
Shadow IT (GenAI tools)	**Stabilize** + **Monitor**	Don't ban it. Catalog what teams are already using. Create a lightweight governance wrapper instead of prohibition
DevOps pipeline	**Compass Rose**	Extend the pipeline to include GenAI model deployment. Low energy, already familiar to teams
Innovation lab	**Compass Rose**	Redirect budget to run 3 GenAI proof-of-concepts with citizen service teams. Increase visibility
Training academy	**Trigger**	Launch a "GenAI for public servants" curriculum. Partner with an external provider for speed
"Always done it this way"	**Interaction** + **Research**	Pair resistant teams with early adopters. Collect and share success stories through internal channels
Data silos	**Request**	Propose a cross-department data catalog to the IT leadership round. Start with metadata, not migration
Staff council	**Interaction**	Involve early. Frame GenAI as augmentation ("better tools for existing roles"), not replacement. Co-design the training approach
BSI requirements	**Research**	Investigate BSI-compliant GenAI hosting options (on-premise LLMs, sovereign cloud). Present options to Katrin

Direction of Travel

The interventions combine into three portfolios:

1. **Governance & Compliance**: Shadow IT wrapper, BSI-compliant hosting research, staff council co-design

2. **Capability Building**: Training curriculum, innovation lab PoCs, pipeline extension
3. **Culture & Adoption**: Early adopter pairing, success story sharing, data catalog initiative

Each portfolio runs as a series of small, parallel experiments — not one large transformation program. The IT leadership round reviews progress monthly. The counter-factual line is reassessed quarterly: what was unchangeable six months ago may have shifted.

Katrin leaves the workshop with something she did not have before: not a strategy document, but a shared picture of the forces at play and a set of interventions sized to her actual room for maneuver.

Why This Works with AME3

In AME3 terms, the Estuarine Map feeds into the Tournament. Each portfolio becomes an Ambition or a set of Goals on the Enterprise Backlog. The individual interventions become Improvements that Teams pull during a Match. The monthly review follows the Anticipate, Advance and Assess loop. The constraint map ensures the organization does not waste energy fighting granite cliffs but instead focuses on what can actually move.

The Estuarine Map also enforces Overall Optimization[16]: by making all actants visible across the enterprise, the portfolios can be evaluated against the Enterprise Product as a whole — not just the unit proposing the change.

Connection to AME3

AME3 Concept	Estuarine Framework Equivalent
Empirical Control	The energy/time grid assessment and safe-to-fail experiments embody Empiricism[17]: hypothesize, intervene, measure

[16] see: Overall Optimization, 29
[17] see: Empiricism, 191

AME3 Concept	Estuarine Framework Equivalent
Evolution	Direction of travel reflects evolutionary positioning —
Focus	where products and services sit on the Genesis-to-Commodity path
Overall Optimization	Making all actants visible across the enterprise prevents suboptimization — each portfolio is evaluated against the Enterprise Product, not local outcomes
Tournament	The quarterly reassessment of the counter-factual and volatile lines mirrors the Tournament's strategic AAA loop
Match	The cadence within which individual interventions are executed as Improvements by Teams
Improvement	Individual safe-to-fail experiments — the smallest unit of change
Ambition	Estuarine portfolios map to Ambitions — purpose, expected successes, and constraints for an Arena Product
Goal	Specific strategic objectives derived from the Estuarine direction of travel, guiding Teams within a Match cycle

References

- Snowden, D. (2022). *Estuarine Mapping* (first edition). cynefin.io
- Snowden, D. (2023). *Cynefin St David's Day* blog series. cynefin.io
- Snowden, D. (2024). Book chapters on complexity and uncertainty management
- NOAA. *Estuaries: Classifying Estuaries by Geology.* oceanservice.noaa.gov
- cynefin.io/wiki/Estuarine_framework
- cynefin.io/wiki/Constraints

9 Tips for Better Planning and Estimation in the Enterprise

Consider two bike journeys: your daily commute to work versus a cycling adventure through Patagonia. For your commute, you can estimate arrival

time within minutes based on experience. But planning a multi-week expedition through unpredictable terrain, weather, and conditions requires a completely different approach. You need flexibility, contingency plans, and the ability to adapt as conditions change.

Enterprise planning works the same way. Simple or complicated tasks are like the daily commute - predictable and easy to estimate. Complex initiatives are like the Patagonia adventure - requiring adaptive planning strategies that accept uncertainty rather than fight it.

Planning in the Simple and Complicated Space

Back to our bike metaphor: Planning your daily commute is straightforward. Repeat it often enough, collect data on travel times, identify the extreme points (high traffic on special days), and you'll reliably arrive on time for your meetings. Simple and complicated tasks work this way - they're predictable, repeatable, and easily estimated. Once you understand them, they become a no-brainer.

Most enterprises handle simple and complicated work reasonably well. The real challenge lies elsewhere: as markets evolve faster and uncertainty grows, the complex space continuously expands. This is where traditional estimation, prediction, and planning methods break down.

Planning in the Complex Space

The Patagonia adventure represents this complex space - environments where every day brings new terrain, unexpected weather, and unforeseen challenges. This is where simple, time-based planning fails and adaptive approaches become essential.

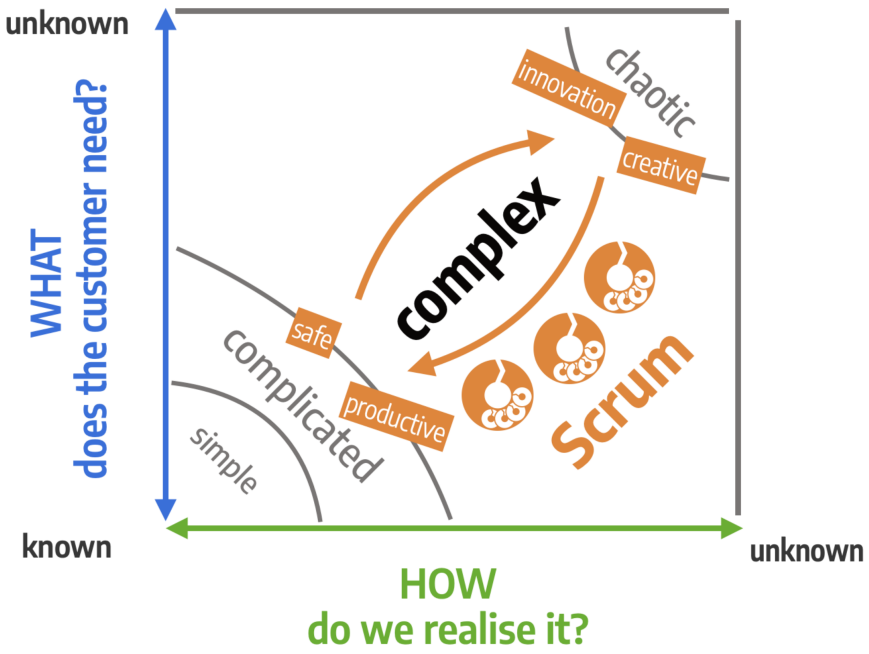


Figure 8.4.: Scrum is designed for the complex space where time-based planning breaks down

The challenges of planning in uncertain environments are not new. The Agile community has developed specific practices for planning and estimating in complex product development - techniques like iterative planning, relative estimation, and velocity-based forecasting that embrace uncertainty rather than pretend it doesn't exist.

> "No plan survives first contact with the enemy."

But long before the Agile Manifesto, military strategists understood the fundamental problem. Helmuth von Moltke[18], a Prussian field marshal, captured this reality in 1871 when he stated: *"No plan of operations extends with any certainty beyond the first encounter with the main enemy forces."*

In business, your "enemy" might be fair competitors in the market, emerging technologies like AI, or simply the inherent complexity of product de-

[18]see: No plan survives first contact with the enemy, 221

velopment. The Agile Manifesto addresses this directly: **responding to change over following a plan.**

This doesn't mean plans have no value. It means we must prioritize adaptability over rigid adherence to predetermined paths.

The more complex the task, the greater the deviation from the plan and the lower the accuracy of estimates. This is not a failure of planning but a characteristic of complex work.

So how do we estimate and plan effectively in this complex space where most enterprises operate today? Here are nine practical tips that challenge conventional wisdom about estimation and planning.

Tip 1: Estimations Are Overestimated

Humans are optimists. Well, that's why we're human. But this creates a cognitive bias in estimation (see Planning Fallacy[19]).

Moreover, the relationship between time spent estimating and estimation accuracy is not linear. It follows a saturation curve. After a certain point, spending more time on estimation doesn't improve accuracy. In fact, it can decrease accuracy as people influence each other.

In short, when someone asks "how accurate is your estimation?" most people answer with a too optimistic number. In reality, it's much worse.

The practical conclusion: invest less time in estimation.

The probability that you're over-investing in estimation and not gaining better accuracy is high. When choosing between detailed estimation techniques or simpler approaches, err on the side of simplicity.

Tip 2: We Seek Security - Planning and Estimating Is Often an Excuse Not to Start

Humans naturally avoid risk. This makes evolutionary sense - why take unnecessary risks? In complex environments where uncertainty is high, this tendency becomes stronger.

[19] see: Planning Fallacy, 222

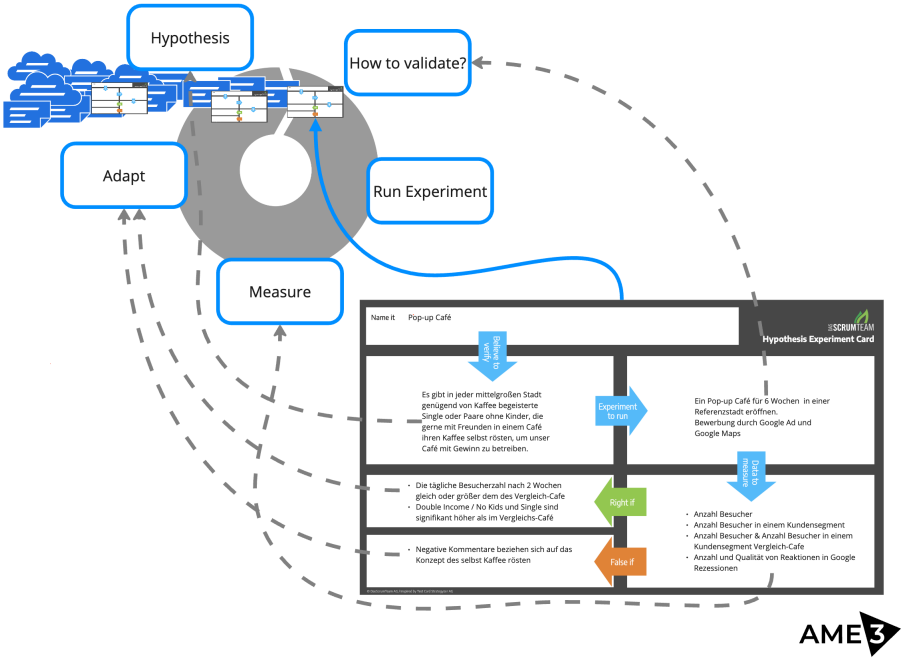
Today's risk in business is typically a difficult conversation with your manager, not physical danger. Yet we still avoid starting.

Several perspectives help overcome this paralysis:

Planning and estimation must be paid for. The time spent on detailed plans and estimates consumes budget. Often, direct implementation would provide more certainty faster.

Only implementation provides absolute certainty. Each implementation validates or invalidates a hypothesis.

Experiments can increase knowledge. Prototypes and spikes can improve planning accuracy. But beware: implementation may still be cheaper than extensive experimentation.



Hypothesis Experiment Cards as Improvements in a Backlog

Use our hypothesis experiment cards when you need more certainty before full implementation. But always ask: would direct implementation with its associated risk actually be less expensive?

Scrum and AME3 are fundamentally empirical control processes. Backlog entries are hypotheses. Advancing in a Match or Sprint generates data and validates them.

Tip 3: Use Different Estimation Methods for Each Level

Don't try to create a single estimation hierarchy from enterprise goals down to individual tasks. This classical project structure plan thinking works in simple environments but fails in complex ones where changes happen too fast.

Instead, use lightweight estimation methods at each organizational level that provide sufficient accuracy with minimal effort:

Enterprise level - Strategic goals and initiatives

- Use relative comparison to past goals, initiatives, or projects
- Consider dimensions like strategic relevance, business benefit, organizational impact, complexity, and CapEx
- Spider diagrams can visualize multi-dimensional comparisons[20] and make relative planning and decision making easier.

[20] see: Visualizing Enterprise Goals, 178

Product/Backlog level - Features and stories

- Relative estimation (Story Points with planning poker or bucket estimation)
- Velocity-based planning
- No estimation (counting items, see Tip 7)

Task level - Daily activities

- Time-based estimates using natural time boxes (days)
- Pull-based planning

Each level has its own reference frame and planning horizon. Don't try to roll up task estimates into story points, and don't try to estimate enterprise

initiatives by detailing them in backlog or project plans.

Tip 4: Choose the Right Reference for Estimation

Relative estimation works because humans find it easier to compare things than to measure them absolutely.

Time is actually a poor reference for many situations:

- How long will a colleague take?
- How long will I take tomorrow when I'm more experienced?
- It's like the ur-meter in Paris changing its length all the time. For a precise measurement, you need an even more precise reference.

Better references include:

Other work items - Planning Poker and bucket estimation compare stories to each other, not to abstract time measures.

Metaphors - Using animals (elephant, cat, mouse) creates mental images. The brain processes relative sizes more naturally than abstract numbers.

Completed work - What we've already delivered provides the most reliable reference frame.

The key is that your reference stays stable even as your team's capability evolves. A size 5 story is always five times larger than a size 1 story, even if the team gets faster at delivering both.

Tip 5: Never Use Estimates for Performance Measurements

What you measure is what you get.

If you measure team performance using velocity based on Story Points, teams will increase their Story Points. Not by working faster, but by estimating higher.

Real example: An organization measured Scrum Masters by their teams' commitment achievement (planned Story Points vs. delivered Story Points). Result? Teams became slower on paper. They planned less per sprint to ensure they always met commitments.

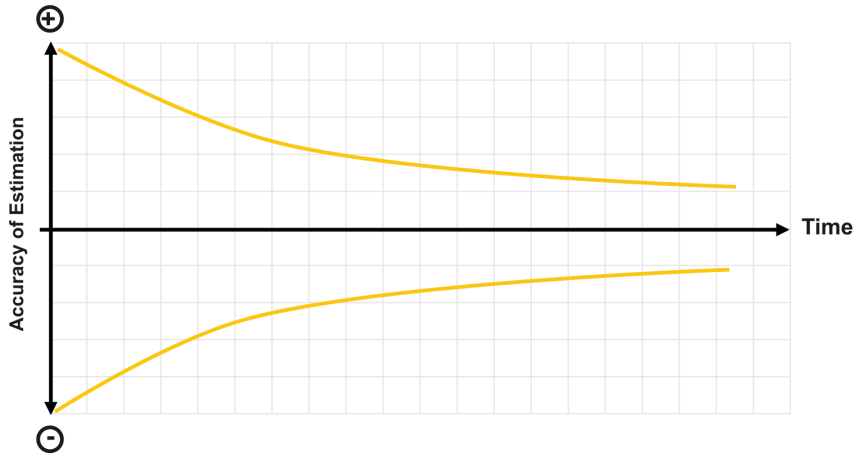
Another anti-pattern: Using velocity as a performance KPI encourages teams to inflate estimates or deliver more features of poor quality that nobody needs.

Estimates should only be used for planning future work, never for performance evaluation, rewards, or punishments.

This is also why time-based estimates are problematic. Humans are paid by time. When you ask for time estimates, you're asking people to estimate something directly tied to their compensation. This creates perverse incentives.

Use estimates for their intended purpose: understanding where you'll be in the future. Don't use them to judge whether teams are "good" or "bad."

Tip 6: Velocity Is the Better Planning Foundation in Complex Environments



Relative estimation with abstract numbers (like Story Points) keeps estimates stable longer than time-based estimates. This allows teams to benefit from increasing accuracy, without constantly reworking their estimates.

Here's why: After three sprints, your team gains experience and becomes more accurate. With time-based estimates, you'd need to rework your entire plan each sprint:

- Sprint 1: Estimated 100 days
- Sprint 2: Actually need 130 days (update all estimates)
- Sprint 3: Actually need only 60 days (update all estimates again)

With relative estimates and velocity:

- Sprint 1: 50 Story Points to release in Backlog, velocity 10 per sprint = 5 sprints
- Sprint 2: 8 Story Points completed. Still 42 Story Points to release, velocity now 8 per sprint = 5.25 sprints
- Sprint 3: 12 Story Points completed. Still 30 Story Points to release, velocity now 12 on average = 2.5 sprints

The estimates don't change. Only the velocity changes. Your five-point item is still five times larger than your one-point item. This makes planning dramatically more stable.

Tip 7: Scrum Has a Built-In Estimation (But Few Know It)

Many people don't realize Scrum has a built-in estimation technique. It's not Planning Poker.

The built-in estimation is pull-based planning.

Think of the childhood game of guessing how many peas fit in a jar. Sprint planning works the same way. The sprint is your jar. Backlog items are your peas. You estimate by pulling items into the sprint until it's full.

This pull-based approach uses a known, fixed timeframe as the reference. A sprint is typically a month or two weeks. We know from experience what we can accomplish in such timeframes.

The modern term for this is "No Estimates" - but it's actually still estimation, just implicit rather than explicit.

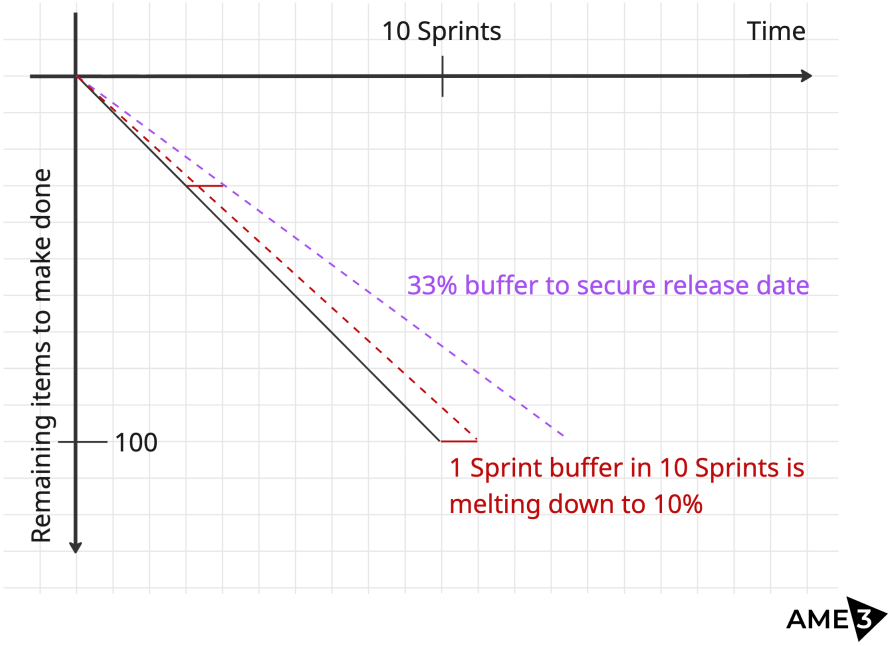
For this to work, you need fixed iterations like Sprints or Matches. Without them, pull-based estimation doesn't function.

Calculate velocity by counting completed items. If Backlog items are sized to fit within a sprint through refinement, they naturally become similar in size. Count them: 10 cards completed in a sprint means your velocity is 10 cards per sprint.

This is simpler for new teams to learn than complex estimation techniques like Planning Poker, and in practice, accuracy is comparable.

Tip 8: Never Plan Absolute Buffers

One of the most common mistakes in planning is using absolute buffers instead of relative ones. The difference is critical.



Consider a burn-down chart where your team has an average velocity of 10. So in 10 sprints it can complete 100 items. If you add a buffer of one sprint to a three-sprint timeline, that's roughly 33% - already optimistic for most projects.

But here's the problem: If you're planning 10 sprints ahead and still use just one sprint as a buffer, you're down to 10% contingency. For 20 sprints, it becomes 5%.

Buffers must scale proportionally with timeline length.

For a 33% buffer, you need:

- 4 sprints of buffer for 10 sprints of work (count only full sprints, so 3.3 becomes 4 sprints)
- 7 sprints of buffer for 20 sprints of work

Remember: that's only 33% contingency, which is actually optimistic for most enterprise projects.

Tip 9: The Value of Planning Lies Not in the Plan Itself, But in the Exchange

> "Plans are worthless, but planning is everything." — Dwight D. Eisenhower[21]

The longer version adds context: *"In preparing for battle I have always found that plans are useless, but planning is indispensable."*

The planning process generates insights. The plan itself is just a snapshot.

This is why techniques like Planning Poker remain valuable even when simpler methods work. The discussion when people show different estimates reveals hidden assumptions and risks. This extreme value analysis surfaces critical misunderstandings.

Planning forces engagement with the future and its uncertainties. The conversation among participants creates shared understanding that no written plan can capture.

If your estimation technique creates good communication containers where teams and leaders discover misalignments and build shared understanding, it's valuable regardless of the accuracy of the estimates produced.

But always ask: could we have this exchange without the estimation technique? If the technique only provides value through communication, perhaps there's a lighter way to achieve the same conversation.

[21] see: Plans are worthless, but planning is everything, 222

Key Takeaways

1. **Invest less time in estimation** - Diminishing returns set in quickly
2. **Implementation provides certainty** - Don't let planning become procrastination
3. **Each level needs its own method** - Don't roll up task estimates to enterprise plans
4. **Choose stable references** - Time changes; relative size doesn't
5. **Never measure performance with estimates** - What you measure is what you get
6. **Velocity beats time estimates** - Keeps plans stable as teams learn
7. **Pull-based planning is estimation** - Scrum has this built in
8. **Use relative buffers, not absolute ones** - Scale contingency with timeline length
9. **Value the process over the plan** - Planning creates shared understanding

Planning and estimation in complex environments requires accepting uncertainty, using lightweight techniques, and focusing on learning over precision. The goal is not perfect accuracy but sufficient confidence to make good decisions.

Related Sources

- Planning Fallacy - The cognitive bias that causes systematic underestimation
- Overconfidence Bias[22] - Why we overestimate our estimation accuracy
- Plans are worthless, but planning is everything
- No plan survives first contact with the enemy

Visualizing Enterprise Goals

Motivation

A network diagram visualizes a strategic initiative, Goal, or project in comparison to other initiatives. At the project portfolio level, Flight Level 1

[22] see: Overconfidence Bias, 221

Kanban, or Enterprise Backlog, a project can be more easily assessed relative to other initiatives and weighted accordingly. This improves decision quality and makes it easier to determine how many initiatives, Goals, or projects can be focused on and worked on simultaneously.

- Only a few pieces of information are displayed that can be quickly processed cognitively.
- The focus is on information relevant to decision-makers.
- Assessment is always relative to other initiatives, making estimates comparable and transparent.

Process

The estimation is conducted by a group of experts. The process can be facilitated using Planning Poker. The assessment is based exclusively on the project description. Estimates can be adjusted in later rounds when new insights or additional experience become available.

> [!Info] The assessment or reassessment takes place in the strategy cycle (Tournament) of AME3 during the Anticipate phase.

Structure

- The dimensions (axes in the spider chart) are scaled from 1 to 20.
- Each dimension should always be estimated in comparison to one or more reference initiatives. The estimation uses a pseudo-Fibonacci sequence: 1, 2, 3, 5, 8, 13, 20.
- To minimize estimation errors, the values of the reference dimensions should ideally be 2, 3, or 5.

Suggested Dimensions:

- **Relevance** in relation to enterprise strategy
- **Benefit** (relative economic value gained from implementation or loss prevented by implementation)
- **Impact** on existing organization, services, and products (e.g., customer outages, process disruptions, effects on other initiatives and projects, and impact on employee productivity)

- **Complexity** (uncertainty regarding success, benefit, technology, etc.)
- **CAPEX** (directly attributable capital expenditures)

Example

Scale 1–20; Pseudo-Fibonacci Estimation: 1, 2, 3, 5, 8, 13, 20

Part IV.

Appendix

9. Sources and Definitions

Acronym AME3

AME3 stands for:

Adaptive **M**etaframework for **E**mpirical **E**nterprise **E**volution

It is pronounced **/e m ri /**

> **Adaptive:** having an ability to change to suit changing conditions.
>
> **Meta-** (prefix): Denoting something of a higher or second-order kind.
>
> **Framework:** A supporting structure around which something can be built
>
> **Empirical:** Based on what is experienced or seen rather than on theory
>
> **Evolution:** A gradual process of change and development
>
> **Enterprise:** 1. a project or undertaking, especially a bold or complex one. 2. a business or company.

A Plan Is Not a Strategy

> A Plan Is Not a Strategy - Roger Martin (YouTube)

In the video "A Plan Is Not a Strategy," Roger Martin explains the distinction between planning and strategy. Planning involves outlining a set of activities or tasks that a company intends to undertake, such as improving customer experience or launching a new product. These activities are often within the company's control and focus on resource allocation. However, they lack the coherence and competitive focus necessary to achieve a strategic goal.

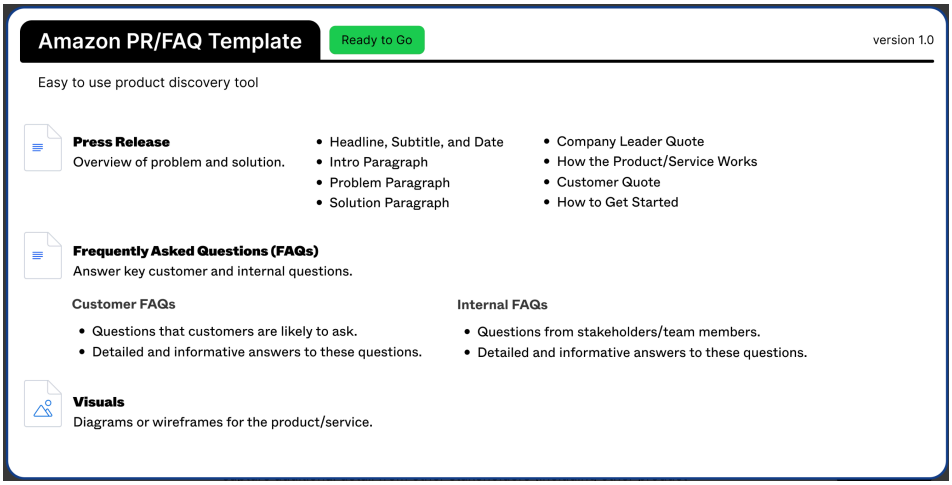
In contrast, strategy is described as an integrative set of choices that positions a company on a chosen playing field to win. It involves a theory about why a company should compete in a particular market and how it can outperform competitors. Strategy requires a coherent plan that can be translated into actions, focusing on achieving a competitive outcome that involves customer engagement and market success. Unlike planning, strategy involves uncertainty and requires managers to accept that they cannot control all variables, but it offers the best chance of achieving significant success.

Amazon's Product Operating Model: A Peek Behind The Curtain, with James Gunaca

Amazon's Product Operating Model - LeSS Talks

The Amazon Product Operating Model is a set of principles, processes, and tools that guide how Amazon teams develop, launch, and manage products. According to James Gunaca, a veteran product manager at Amazon, this model emphasizes customer-centric thinking, iterative experimentation, and strong documentation practices like the PRFAQ. It balances standardized processes with flexibility, allowing teams to adapt strategies to their specific context while maintaining accountability and a focus on business outcomes.

Example Structure PRFAQ



Agile

> Manifesto for Agile Software Development

The Agile Manifesto is a foundational declaration establishing core values for software development, created in 2001 by 17 industry pioneers including Kent Beck, Mike Beedle, Martin Fowler, Ken Schwaber, and Jeff Sutherland. It prioritizes four key principles: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. The manifesto connects to the Twelve Principles of Agile Software and has been translated into 70+ languages, reflecting its global adoption.

AME3

AME3 (formerly named GAME3) provides a framework to lead the evolution of a business's products and services.

AME3 stands for **A**daptive **M**etaframework for **E**mpirical **E**nterprise **E**volution. With AME3 an enterprise can adopt an operational model

based on three pillars: a Leadership System, a Strategy for Evolution, and Enterprise-wide Rules.

AME3 is fostering the use of popular Agile and Lean methods like Scrum, LeSS and Kanban in combination with decision frameworks like Cynefin and Wardley Mapping. It therefore stands on the shoulders of giants.

The framework is designed primarily for small to medium-sized enterprises (SMEs). However, larger enterprises can also implement it within sufficiently autonomous divisions.

The idea of AME3 started to evolve in 2004, when Andreas Schliep and Peter Beck were part of one of the earliest scaled Scrum-based organizations. Over the course of numerous experiments conducted within their own company, consulting engagements with clients, and extensive collaboration with the Agile, Lean, and Scrum communities, the concept of AME3 emerged into a concrete picture.

Artificial Life

> An Introduction to Artificial Life for People who Like AI - The Gradient

Artificial Life, often referred to as ALife, is the scientific study of life through the creation of artificial systems that exhibit behaviors characteristic of natural living systems. This field, as defined by Christopher Langton, involves a bottom-up approach to understanding the fundamental principles of life by building living systems from scratch. Researchers like Lana Sinapayen explore how these systems can mimic aspects of known biological life, focusing on general principles such as emergence, information, and computation.

Also: Open Ended Evolution

Chief Executive Officer (CEO)

> Chief executive officer - Wikipedia

A Chief Executive Officer (CEO) is the highest-ranking corporate executive in an organization, such as a company or a nonprofit organization.

Build-Measure-Learn

From The Lean Startup by Eric Ries

The Build-Measure-Learn feedback loop is a core principle of lean startup methodology introduced by Eric Ries. It is a cyclical learning process that prioritizes speed in product development through three sequential phases: transforming ideas into a minimum viable product (Build), measuring customer reactions and market performance (Measure), and learning from results to determine whether to continue or pivot direction (Learn). This iterative approach reduces market risk by enabling entrepreneurs to test business assumptions early with minimal resources, rather than relying on extensive planning or large initial investments before launch.

Brooks's Law

Brooks's law - Wikipedia

Brooks's law, coined by Fred Brooks in his 1975 book *The Mythical Man-Month*, is an observation in software project management stating that "Adding manpower to a late software project makes it later." This principle highlights the challenges of increasing team size on delayed projects, often leading to further delays due to factors such as training and communication overhead.

Complex Adaptive System

Complex adaptive system - Wikipedia

A Complex Adaptive System (CAS) is a dynamic network of interactions where the behavior of the whole system is not easily predictable from the behavior of its individual components. It is characterized by its ability to adapt and self-organize in response to changes in the environment.

John H. Holland describes CAS as systems with numerous components, often referred to as agents, that interact, adapt, or learn. These systems are found in various domains, including natural and social sciences, and exhibit properties like self-similarity, emergence, and resilience.

Complex environments are dynamic in nature: cause and effect relationships can only be deduced in retrospect.

Complex System

> Complex system - Wikipedia

A complex system is a system composed of many interacting components, which may include dependencies, competitions, and relationships. These interactions make the system intrinsically difficult to model. Examples include Earth's climate, the human brain, and social organizations. Complex systems exhibit properties such as nonlinearity, emergence, and adaptation.

They are studied across various fields, including physics, biology, and economics, to understand how relationships between parts give rise to collective behaviors.

Conway's Law

> How Do Committees Invent? by Melvin E. Conway (1968)

Conway's Law states that **organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations.**

The paper's conclusion emphasizes:

> The basic thesis of this article is that **organizations which design systems (in the broad sense used here) are constrained to produce designs which are copies of the communication structures of these organizations.** We have seen that this fact has important implications for the management of system design. Primarily, we have found a criterion for the structuring of design organizations: a design effort should be organized according to the need for communication.
>
> This criterion creates problems because the need to communicate at any time depends on the system concept in effect at that

time. Because the design which occurs first is almost never the best possible, the prevailing system concept may need to change. **Therefore, flexibility of organization is important to effective design.**

Cynefin

About Cynefin Framework - The Cynefin Co

The Cynefin Framework is a leadership decision-making tool distinguishing operational domains to match responses with contextual reality. Developed by Dave Snowden in 2005, it emphasizes sense-making to differentiate between complex and routine challenges, preventing inappropriate problem-solving approaches. Applied across sectors from healthcare to international development, the framework helps organizations avoid both overthinking simple issues and oversimplifying complex ones, enabling more effective and contextually appropriate decision-making.

Daily Scrum

Daily Scrum - The Scrum Guide

The Daily Scrum is a 15-minute event held every working day of the Sprint, designed for the Developers of the Scrum Team to inspect progress toward the Sprint Goal and adapt the Sprint Backlog as necessary. This meeting fosters communication, identifies impediments, and promotes quick decision-making. It is not limited to a specific structure, allowing Developers to choose techniques that best focus on progress and create an actionable plan for the next day. The concept of the Daily Scrum is part of the Scrum framework developed by Jeff Sutherland and Ken Schwaber.

Deduction

Deduktion - Wikipedia

Deduction, derived from the Latin term "deductio," refers to the process of drawing logically necessary conclusions from given premises. A conclusion is considered deductively valid if it logically follows from the premises, meaning the truth of the premises guarantees the truth of the conclusion. Alfred Tarski highlights that logical consequence, a key aspect of deduction, is necessary, formal, and a priori recognizable. Deductive reasoning is studied in various fields, including logic, psychology, and cognitive sciences, each emphasizing different aspects of the process.

Climatic Patterns by Simon Wardley

> Exploring the Map - Wardley Maps

Climatic Patterns, as defined by Simon Wardley, are recurring forces or trends in business environments that shape the evolution of components within a value chain, regardless of individual actions. These patterns include economic shifts, technological advancements, and competitor behaviors that drive change and influence strategic decisions.

An overview can also be found here

Drive, by Dan Pink

> Drive: The Surprising Truth About What Motivates Us - Dan Pink

"Drive" is a bestselling book written by Daniel H. Pink. It presents a groundbreaking perspective on motivation, challenging the traditional carrot-and-stick approach. Pink argues that the key to high performance and satisfaction in work, school, and home lies in our innate need to direct our own lives, learn and create new things, and contribute positively to the world. Drawing from four decades of scientific research on human motivation, Pink highlights the gap between what science knows and what business does. He identifies autonomy, mastery, and purpose as the three elements of true motivation and provides practical techniques for implementing these elements. The book has been praised by various reviewers, including Kirkus, Miami Herald, and Financial Times, for its insightful and transformative approach to motivation.

Dunbar's Number

Dunbar's number - Wikipedia

Dunbar's number is a concept introduced by British anthropologist Robin Dunbar. It suggests that humans can maintain stable social relationships with about 150 people. This number represents the cognitive limit of how many individuals one can know personally and interact with regularly.

Ecosystems

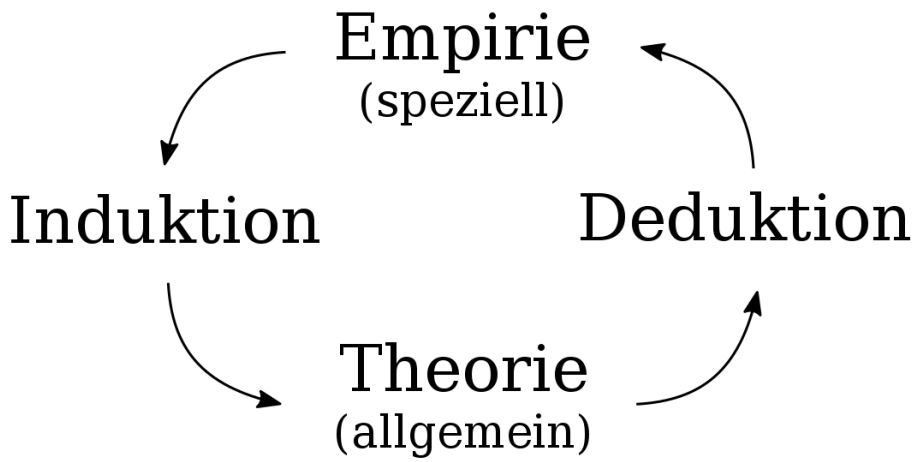
Ecosystems - Wardley's Blog

An ecosystem, as described by Simon Wardley, is a network of interconnected actors—such as companies, individuals, and technologies—that interact and depend on each other within a business environment. These interactions create shared value and enable rapid adaptation to change. Ecosystems are not just collections of parts, but dynamic systems where collaboration, competition, and co-evolution drive innovation and growth.

Empiricism

Empirie - Wikipedia (German)

Empiricism is a philosophical approach that emphasizes the role of sensory experience in the formation of knowledge. Originating in the 17th century, it is closely associated with philosophers such as Francis Bacon and David Hume. Empiricism asserts that all knowledge is dependent on experience and observation, contrasting with rationalism, which emphasizes reason and innate ideas.



Doctrine by Simon Wardley

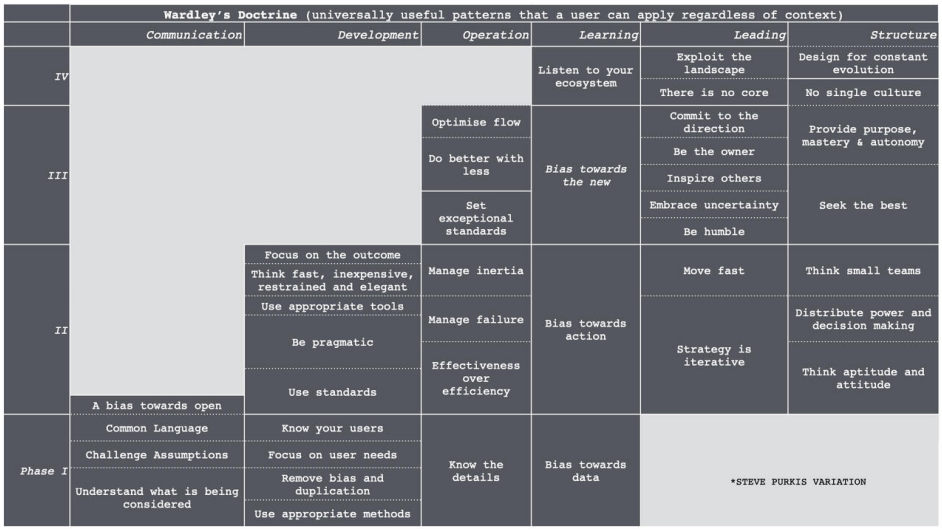
Doctrine Assessment Tool - Learn Wardley Mapping

Doctrine, as defined by Simon Wardley, refers to a set of universally applicable principles that guide decision-making and actions within an organization. These principles are not specific to any particular context but are meant to be broadly applicable across various situations. Wardley's doctrine emphasizes the importance of understanding the landscape, anticipating changes, and adapting strategies accordingly to ensure organizational success.

Wardley's Doctrines Table

Wardley's Doctrines Table is a visual reference framework that systematizes Simon Wardley's universally applicable principles for organizational decision-making and strategic planning. The table organizes doctrines into categories such as communication, development, operation, learning, leading, and structure, providing a comprehensive checklist for assessing organizational maturity. Originally developed as part of Wardley Mapping methodology, these doctrines serve as context-independent guidelines

that help organizations understand their landscape, anticipate change, and adapt strategies accordingly.



Enterprise Scrum

> Enterprise Scrum Introduction - Mike Beedle

Enterprise Scrum, as defined by Mike Beedle, is an adaptation and extension of Scrum that utilizes abstraction, generalization, and parameterization. It is designed to be a scalable and generic framework for any management purpose, allowing organizations to remain agile at all levels and across various domains.

Tragically, Mike Beedle lost his life in Chicago in 2018 during an apparent robbery, preventing him from further advancing his work on Enterprise Scrum.

In some respects, AME3 builds upon the foundation established by Enterprise Scrum. However, there are significant differences, largely because AME3 was developed nearly eight years after Mike Beedle's work, allowing us to incorporate broader community experience and our own insights.

Comparing AME3 and Enterprise Scrum

The concept of applying Scrum as a framework to manage an entire company dates back to the early days of Scrum itself. This idea inspired us to found DasScrumTeam in 2010, an organization fully operated using Scrum. More and more organizations have adopted this approach.

Enterprise Scrum captured this idea explicitly by generalizing Scrum's concepts—such as roles, ordered work, and iterative cadence—to any business context.

AME3 follows the same spirit, but adds a clearer enterprise structure, an explicit leadership system (Owner, System Lead, Team), and strategic doctrines to guide long-term evolution.

AME3 does not focus solely on Scrum; within an Arena, a variety of methods and frameworks can be used. In addition to Scrum, approaches like Kanban, LeSS, or custom team-based methods are also supported.

At the Enterprise level, methods like Wardley Mapping are an option but not mandatory. This flexibility is one reason why AME3 considers itself a Metaframework.

What AME3 and Enterprise Scrum have in common

- Leadership functions align in purpose and accountability: Enterprise Scrum defines a Business Owner, a Coach, and a Team; AME3 defines Owner, System Lead, and Team. In both, the owner role steers for value, the coach/system lead enables an effective work system, and the team executes and improves.
- Ordered work and value focus: Enterprise Scrum works from a generic Value List with Value List Items (VLIs) prioritized to maximize business value; AME3 works from ordered Enterprise Backlog and Arena Backlog with Goals and Improvements prioritized for customer and business impact.
- Empirical, iterative cadence: Enterprise Scrum runs iterative Cycles (PC3R) and supports recursive cadence across levels; AME3 runs nested Matches (Arena cadence) and Tournaments (enterprise cadence), grounded in the Empirical Product Control Loop.

- Pull and self-management: In both, teams pull a feasible amount of work into each cycle, commit to getting it Done against clear rules of completion, and self-organize to deliver value.
- Multi-level applicability: Enterprise Scrum instances and canvases generalize Scrum to any domain and scale; AME3 explicitly structures work at Enterprise and Arena levels with consistent rules and shared cadence.
- Continuous improvement: Both emphasize regular inspection and adaptation of the work and the way of working to increase effectiveness and value.

The Differences

- Structure and scope: AME3 prescribes two explicit levels (Enterprise and Arena) with defined artifacts, constraints, and leadership functions (e.g., Enterprise Product, Arena Product, Match, Tournament). Enterprise Scrum provides generic instances and templates that can be configured for many domains without prescribing an enterprise/arena split.
- Terminology and artifacts: Enterprise Scrum uses generalized terms such as Vision, Value List, Value List Items (VLIs), Cycles, and canvases. AME3 uses product-oriented artifacts and backlogs (e.g., Enterprise Backlog, Arena Backlog) and names cadence explicitly as Matches and Tournaments.
- Separation of accountabilities: AME3 makes separation explicit (e.g., the Enterprise Owner cannot be a System Lead or Enterprise System Lead; the System Lead focuses on the effectiveness of the work-system). Enterprise Scrum's Coach and Business Owner are roles within the team configuration and the definition does not impose AME3's specific separations.
- Strategic doctrines: AME3 embeds explicit doctrines such as Empirical Control, Evolution Focus (and Overall Optimization) to guide enterprise evolution. Enterprise Scrum focuses on business agility outcomes and configuration guidance rather than naming such doctrines.
- Cadence naming and nesting: AME3 standardizes shared cadence across Arenas via Matches/Tournaments; Enterprise Scrum emphasizes recursive/nested Cycles and configurable canvases for different business contexts.

Further Reading and References on Enterprise Scrum

Field guide to managing complexity (and chaos) in times of crisis

> Field guide to managing complexity (and chaos) in times of crisis - Cynefin Wiki

The "Field Guide to Managing Complexity (and Chaos) in Times of Crisis" is a practical resource for decision-makers, inspired by the Cynefin® framework. Authored by Snowden, D. and Rancati, A., it provides strategies for navigating crises through a four-stage approach. The guide emphasizes setting boundaries, building informal structures, and maintaining flexibility. It includes real-life examples and action items to support the theoretical framework, aiming to enhance decision-making during turbulent times.

Field-programmable Photonic Nonlinearity

> Field-programmable photonic nonlinearity - Nature Photonics

Field-programmable photonic nonlinearity is a technology that allows the dynamic and precise control of nonlinear optical responses within integrated photonic devices. Developed by Tianwei Wu, Yankun Li, Li Ge, and Liang Feng, this approach enables in situ training and reconfiguration of photonic networks, making it possible to implement complex activation functions commonly used in artificial intelligence. This innovation extends the programmability of photonic systems beyond the linear domain, offering new possibilities for efficient and flexible reconfigurable computing.

Flight Levels

> Flight Levels - LEANability

Flight Levels is a framework for scaling agile and Kanban across organizations, developed by Klaus Leopold. It organizes work and coordination across three distinct levels: operational (team level), coordinating (cross-team coordination), and strategic (portfolio and strategic direction). The

framework helps organizations visualize work flows, dependencies, and bottlenecks across different organizational layers. Flight Levels emphasizes evolutionary change, interaction between levels, and making work visible across the entire system to improve flow and delivery effectiveness.

Generative Artificial Intelligence

> Wikipedia - Generative artificial intelligence | Merriam-Webster - Generative AI

Generative Artificial Intelligence (GenAI) is a subfield of artificial intelligence that uses machine-learning models to produce new content, including text, images, video, audio, and code, in response to user prompts. Early foundations include Markov chains (1906) and Harold Cohen's AARON (1970s). The field advanced significantly with deep learning breakthroughs: variational autoencoders, Generative Adversarial Networks (GANs, 2014), and the Transformer architecture (Vaswani et al., 2017). The Transformer enabled Large Language Models (LLMs) like GPT and BERT (2018), which became the dominant architecture for text generation. The mainstream AI boom began with DALL-E and Midjourney (2020-2022) for images, and accelerated with ChatGPT's release in November 2022. By 2025, GenAI models like GPT-4, Claude, Gemini, and DeepSeek had expanded from text generation to complex reasoning, code generation, and autonomous agent capabilities.

Higher-Order System

A **higher-order system** is a system characterized by interactions, relationships, or dynamics that transcend simple pairwise or first-order connections, encompassing more complex structures, dependencies, or hierarchies among its components. In the context of system science, mathematics, and network theory, a higher-order system may refer to one where the behavior, evolution, or organization arises from interactions involving three or more elements simultaneously, or from nested subsystems forming higher levels of organization. Such systems are described using frameworks that capture their multidimensional or multilevel structure, such as hypergraphs, hypermatrices, or higher-order differential equations. Higher-order systems are

crucial for modeling collective phenomena, emergent behaviors, and complex dependencies not reducible to lower-order interactions[12345].

Definition in the Context of Wardley Mapping: In Wardley Mapping, "higher-order system" describes an emergent structure or value-creating capability that is enabled by the commoditization and ubiquity of underlying components in the value chain. The identification and anticipation of higher-order systems are central to strategic advantage, innovation, and navigating cycles of change in business ecosystems[678].

References

How to Organise Your Teams

How to organise yourself? A dangerous path - Wardley's Blog

Organizing teams effectively involves adapting to constant change by implementing the Explorers, Villagers, and Town Planners (EVTP) system, as described by Simon Wardley. This approach requires understanding the evolving nature of tasks and aligning team mindsets and skillsets accordingly. The process includes improving situational awareness, introducing an Intelligence Function for mapping and challenging projects, and eventually restructuring the organization into cells based on attitudes and aptitudes.

---

[1]Higher-order systems. In *Handbook of Geometric Constraint Systems Principles* (Springer, 2022). Read PDF

[2]Benson, A. R. (2023). What are higher-order networks?. *SIAM Review, 65*(2), 251-272. Read abstract

[3]Qi, L., & Ye, Y. (2023). Hypermatrix algebra and irreducible arity in higher-order systems: Concepts and perspectives. *Mathematics in Computer Science, 17*(1), 109-134. Read abstract

[4]Gundlach, C., & Martín-García, J. M. (2015). Hyperbolicity of high-order systems of evolution equations. *International Journal of Modern Physics D, 24*(12), 1550001. Read abstract

[5]Iacopini, I., Petri, G., & Barrat, A. (2024). Contagion dynamics on higher-order networks. *Nature Reviews Physics, 6*, 255–267. Read article

[6]Simon Wardley, *Wardley Maps - Coach agile* (2020), PDF

[7]Simon Wardley, "Open source as weapon", *Bits or Pieces* (2015), blog

[8]Simon Wardley, "I don't like Hayek vs Keynes", *Bits or Pieces* (2012), blog

This method aims to enhance adaptability and efficiency in a dynamic environment.

See also: Wardleys Doctrines Table

Increment

> The Scrum Guide - Increment

An Increment is a concrete stepping stone toward the Product Goal in Scrum. It represents a usable, verified deliverable that is additive and cumulative with all prior Increments. Each Increment must meet the Definition of Done to ensure quality and usability. Multiple Increments may be created within a single Sprint and can be delivered to stakeholders before the Sprint concludes. The sum of Increments presented at the Sprint Review supports empirical inspection and adaptation, treating each Sprint as a measurable delivery of value rather than merely a planning period. This reinforces transparency and continuous progress.

Induction

> Induktion (Philosophie) - Wikipedia

Induction, as defined by philosophers like Aristotle, is the process of drawing general conclusions from specific observations. It contrasts with deduction, which derives specific conclusions from general premises. Hume argued that induction cannot lead to necessary and universal laws, while theorists like Reichenbach and Carnap attempted to formalize inductive reasoning. Popper critiqued induction, suggesting it is an illusion and advocating for a deductive approach. Induction remains a topic of debate in philosophy, logic, and cognitive sciences.

Kanban

> Kanban (development) - Wikipedia

Kanban is a Lean method for managing and improving work across human systems by visualizing workflow, balancing demands with available capacity, and addressing bottlenecks. Originating from the Toyota Production System in the late 1940s, Kanban was adapted for knowledge work and software development by authors such as David Anderson (2010) and Corey Ladas (2008). It uses visual tools like the Kanban board to visualize work and workflow, and emphasizes limiting work in progress (WIP) to enhance efficiency and predictability. The methodology enables teams to pull work as capacity allows, establish explicit policies, and implement continuous improvement through feedback loops.

Claude Flow

Claude Flow on GitHub

Claude Flow is an enterprise-grade AI orchestration platform designed to coordinate multi-agent systems for intelligent software development workflows. Maintained by ruvnet on GitHub with over 9.5k stars, it features 25 Claude Skills for natural language-activated development capabilities, swarm orchestration with hive-mind architecture, and 100 MCP tools for automated workflow management. The platform includes hybrid memory systems combining AgentDB (96x-164x faster vector search) and ReasoningBank for pattern matching, achieving an 84.8% solve rate on SWE-Bench with 2-3ms query latency. It offers native integration for Claude Code via the MCP protocol and advanced hooks for pre/post-operation automation.

Larman's Laws

Larman's Laws of Organizational Behavior

Larman's Laws of Organizational Behavior describe how established organizations resist structural change to preserve existing management hierarchies and power dynamics. Developed by Craig Larman through decades of organizational consulting, these empirical observations outline five patterns: organizations optimize to maintain the status quo, redefine new initiatives to mean the same thing, dismiss genuine change efforts as impractical, co-opt

displaced managers as coaches, and operate on the principle that "culture follows structure" in large organizations rather than the reverse.

Lean

> Lean manufacturing - Wikipedia

Lean Management, often referred to as Lean Manufacturing, is a systematic method for waste minimization within a manufacturing system without sacrificing productivity. Originating from the Japanese manufacturing industry, it takes into account waste created through overburden and waste created through unevenness in workloads.

Lean Management aims to create more value for customers by utilizing fewer resources. A lean organization understands customer value and focuses its key processes to continuously increase it. The ultimate goal is to provide perfect value to the customer through a perfect value creation process that has zero waste.

To accomplish this, lean thinking changes the focus of management from optimizing separate technologies, assets, and vertical departments to optimizing the flow of products and services through entire value streams that flow horizontally across technologies, assets, and departments to customers.

Eliminating waste along entire value streams, instead of at isolated points, creates processes that need less human effort, less space, less capital, and less time to make products and services at far less costs and with much fewer defects, compared with traditional business systems.

LeSS

> LeSS - Large-Scale Scrum

LeSS, or Large Scale Scrum, is an organizational system designed for scaling Scrum, lean, and agile development to large product groups. Developed and shared by The LeSS Company B.V. since 2005, it leverages their extensive experience and knowledge to help organizations succeed in scaling their operations.

What is the Purpose of Life?

What is the Purpose of Life?

This video explores the connection between life, energy, and entropy. Scientists such as Michael Russell and Albert Szent-Györgyi describe life as a process that increases entropy, even though living organisms are highly organized. The video explains how energy from the Sun is gradually degraded as it moves through plants, animals, and cells, with each step increasing entropy and reducing the amount of useful energy.

The process begins with photosynthesis. Plants capture solar energy and store it as sugar. Animals eat the sugar and convert it into ATP, which powers muscles and repairs cells. However, with each transformation, some energy is lost as heat. This demonstrates that life helps the universe move toward greater disorder, or higher entropy.

The video also suggests that life may have started as complex chemical reactions that could use available energy and increase entropy, possibly in environments like ocean vents. It draws a parallel with stars, which also increase entropy by converting hydrogen into helium and releasing energy. In summary, both life and stars contribute to the universe's tendency to increase entropy, and in a sense, life continues the work started by the stars.

This video is part of a series on time and entropy, inspired by Sean Carroll's book "The Big Picture," which is available on Audible.

Mike Beedle

Mike Beedle - Wikipedia

Mike Beedle was an American software engineer and theoretical physicist, known for co-authoring the Agile Manifesto and pioneering the Scrum framework. He was instrumental in the early adoption and implementation of Scrum, collaborating with Ken Schwaber and Jeff Sutherland. Beedle also developed the concept of Enterprise Scrum, promoting its use for scaling Scrum practices across organizations.

Manifesto for Agile Software Development

Manifesto for Agile Software Development

The Manifesto for Agile Software Development is a foundational declaration created in 2001 by seventeen software development pioneers including Kent Beck, Martin Fowler, Ken Schwaber, Jeff Sutherland, and Ward Cunningham. It established four core values emphasizing individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. While acknowledging the value of traditional approaches, the manifesto fundamentally reoriented the industry toward iterative development, continuous feedback, and human-centered practices that remain influential in shaping contemporary development culture globally.

Nigel Scale

The Nigel Scale

The Nigel Scale, as defined by Nigel Baker, is a tool used in Scrum discussions to differentiate between Scrum rules, best practices, and non-Scrum practices. It's divided into three categories: NS1, NS2, and NS3. NS1 represents the hard rules of Scrum, such as Sprints, Daily Scrums, and ScrumMasters. NS2 refers to best practices within Scrum, which can vary based on the team's needs and the specific situation. NS3 denotes practices that are not part of Scrum or are considered anti-practices. The Nigel Scale is used to guide decision-making and ensure clarity in Scrum implementation.

OODA Loop

OODA loop - Wikipedia

The OODA Loop, developed by United States Air Force Colonel John Boyd, is a decision-making model that stands for Observe, Orient, Decide, and Act. It is used to describe a continuous cycle of feedback and observations

that enable agility and effective decision-making in various fields, including military strategy, business, and law enforcement. The model emphasizes the importance of processing this cycle quickly to gain an advantage over opponents by reacting to events more rapidly and effectively.

Obeya

> Obeya - Wikipedia

Obeya (from Japanese *Ōbeya* "large room") is a team-based management tool, originating from Japanese lean manufacturing, designed to improve collaboration, decision-making, and problem-solving at an administrative level. It involves bringing key stakeholders together in a dedicated space—physical or virtual—where visual information is shared to align strategy and execution. The concept and its 11 guiding principles have been developed and promoted by experts such as Dolf Reijnders and Bart Bongers of the Obeya Association.

Open Ended Evolution

> An Introduction to Artificial Life for People Who Like AI - The Gradient

Open Ended Evolution (OEE) is a concept in Artificial Life research that refers to systems capable of generating increasing complexity over time, without a predefined endpoint. Kenneth O. Stanley, a prominent researcher in the field, describes OEE as the process where systems evolve to become more complex, akin to the way life on Earth has developed. The goal is to create artificial systems that start from simple beginnings and evolve into highly complex entities, potentially leading to the emergence of intelligent systems.

> OEE is this idea that some systems get exponentially more complex with time, and that complexity never ceases to increase. Life on Earth is said to be such an open ended system. Creating OEE in a computer or in a chemical system would mean that you start from something simple, maybe a soup of molecules, or

an empty simulation, and get immense complexity out of it, like living animals or maybe conscious beings.

PDCA Cycle

PDCA - Wikipedia

The PDCA Cycle, also known as the Plan-Do-Check-Act cycle, is an iterative design and management method used for controlling and continually improving processes and products. Originating from physicist Walter A. Shewhart and later modified by W. Edwards Deming, the cycle involves planning objectives, executing them, evaluating the results, and making necessary adjustments. It emphasizes observation, critical thinking, and iterative learning to enhance quality and efficiency in business operations.

Paul Watzlawick

Paul Watzlawick - Axiome der Kommunikation

Paul Watzlawick (1921-2007) was an Austrian-American communication theorist and psychologist who developed five pragmatic axioms explaining fundamental principles of human communication. His work bridges psychology and communication studies, offering practical frameworks for understanding relationship dynamics. His most famous insight, "One cannot not communicate," emphasizes that all behavior conveys meaning. Watzlawick's axioms address how communication is unavoidable, contains content and relational dimensions, operates in circular patterns, uses both verbal and nonverbal modes, and creates symmetrical or complementary relationships. His theories remain influential in family therapy, organizational communication, and conflict resolution.

People Are Hard, Code is Easy

Linus Torvalds on YouTube

Process Dynamics, Modeling, and Control

> Process Dynamics, Modeling, and Control - Oxford University Press

Process Dynamics, Modeling, and Control is a comprehensive chemical engineering textbook published in 1994 by Oxford University Press as part of the Topics in Chemical Engineering series. Authored by Babatunde Ogunnaike (Senior Research Associate at E.I. duPont de Nemours and Adjunct Professor at University of Delaware) and W. Harmon Ray (Steenbock Professor of Engineering at University of Wisconsin), the book offers a modern view of process control with a unified approach to model representations, process model formation and identification, multivariable control, statistical quality control, and model-based control.

Projekt Lightspeed Der Weg Zum BioNTech-Impfstoff - Und Zu Einer Medizin Von Morgen

> Projekt Lightspeed: Der Weg zum BioNTech-Impfstoff - Amazon

Projekt Lightspeed chronicles the development of the BioNTech COVID-19 vaccine by founders Uğur Şahin and Özlem Türeci, with Joe Miller. Published in German, the book documents the scientific journey, organizational challenges, and breakthrough moments that led to creating a vaccine in record time during the pandemic. It combines insider perspectives from company founders with accessible explanations of mRNA technology and the vaccine development process, offering insights into modern medicine's rapid innovation potential and the future of personalized treatments.

Projekt Lightspeed

Name of the project to develop the COVID-19 vaccine at BioNTech. Described in the book Projekt Lightspeed Der Weg zum BioNTech-Impfstoff - und zu einer Medizin von morgen

DasScrumTeam

DasScrumTeam

DasScrumTeam is a group of experienced Agile and Scrum experts, founded by Andreas Schliep and Peter Beck, who have been pioneers of Scrum in Germany, Austria, and Switzerland since 2004. The Team includes Certified Scrum Trainers™, Certified Enterprise Coaches™, and Agile Coaches who support organizations and individuals in enhancing their ways of working with Scrum, Kanban, and Agile principles. DasScrumTeam is fully structured around Scrum and consistently applies the same Agile practices they teach to their clients.

SAFe

Scaled Agile Framework

The Scaled Agile Framework (SAFe), developed by Scaled Agile, Inc., is a knowledge base of proven, integrated principles and practices for Lean, Agile, and DevOps.

SAFe is not without critics in the Agile community. See also The SAFe Delusion.

ScALeD

ScALeD Principles

ScALeD, or Scaled Agile Lean Development, is a set of guiding principles designed to help organizations effectively scale their agile practices. It emphasizes the importance of excited customers, happy and productive employees, global optimization, supportive leadership, and continuous improvement. The principles were formulated by Christoph Mathis, Markus Gärtner, Stefan Roock, Andreas Schliep, and Peter Beck.

How AME3 Implements the ScALeD Principles

AME3 was designed by two of the ScALeD co-authors — Andreas Schliep and Peter Beck — to put these principles into practice. The framework provides concrete leadership roles, organizational structures, and iterative rhythms that bring each ScALeD principle to life at enterprise scale.

Excited Customers. AME3 places customer value at the center through clear accountability. The Owner defines the Ambition — why an Arena exists and what success looks like — and selects Goals from the Enterprise Backlog to focus the work. Teams deliver small, working Improvements every Match, building a growing Arena Product that customers and stakeholders can inspect. Because Matches are timeboxed to one month or less, the organization gets fast feedback on whether it is building the right thing and can adjust direction before wasting effort.

Happy and Productive Employees. Each Arena contains self-managing, cross-functional Teams that decide how to accomplish their work. Teams pull Improvements from the Arena Backlog, coordinate with each other directly, and define their own completion standards. The Coach develops team capabilities and facilitates decision-making without directing. This combination of autonomy, growing mastery, and clear purpose — provided through a shared Goal — creates an environment where people can do their best work.

Global Optimization. AME3's strategic doctrine Overall Optimization ensures that every improvement benefits the entire enterprise, not just the unit making the change. All Teams within an Arena share the same Goal, so their work aligns naturally without competing individual targets. The Arena Product and Enterprise Product are visible to everyone, and the Tournament provides regular enterprise-wide inspection by Accountable Representatives. Matches and Tournaments create synchronized rhythms where all parts of the organization coordinate and reflect together. Because organizational structure and product architecture are two sides of the same coin, AME3 treats them as one problem — slicing organizations along stable interfaces and reducing dependencies systematically.

Supportive Leadership. AME3 distributes leadership across three complementary functions: the Owner leads toward product success, the Coach leads toward effective work systems, and the Team leads toward customer

satisfaction. None of these functions can override the others, which prevents top-down command-and-control. The Owner sets direction through Ambitions and Goals, not daily instructions. At the enterprise level, Accountable Representatives assess overall progress and advise the Enterprise Owner, while the Enterprise System Lead supports organizational evolution. Leadership's job is to provide clarity, support, and the right environment — decisions are made by those doing the work.

Continuous Improvement. AME3 embeds inspection and adaptation at two levels. Within each Match, Teams assess the Arena Product and their own work process, then feed new Improvements into the Arena Backlog for the next cycle. At the Tournament level — typically quarterly — Accountable Representatives inspect the Enterprise Product and organizational design, and the Enterprise Owner adjusts Ambitions and the Enterprise Backlog accordingly. This dual-loop structure means that product quality, work processes, and organizational design all improve continuously based on evidence, not assumptions.

Scaling Complex Systems by Building on Agile Frameworks with Hexi

Scaling Complex Systems - Scrum.org Webinar Slides (PDF)

Scaling Complex Systems by Building on Agile Frameworks involves using a 'best of breed', multi-method approach to agility. This approach goes beyond simple, defined processes and embraces co-evolutionary and adaptive methods to enhance the flow of value. It is based on complex adaptive systems theory, which views scaling as a process of decomposition and recombination, similar to DNA. This method, known as Hexi, was discussed by Prof Dave Snowden, creator of the Cynefin framework, and Nigel Thurlow, co-creator of The Flow System and creator of Scrum The Toyota Way, in a Scrum Pulse webinar.

Key Points:

- Complex systems scale by decomposition (to the lowest level of coherent granularity) and recombination

- Granularity at which people can make sense of things without specific technical knowledge.
- Complex systems do not scale by imitation or replication or through case-based approaches.
- CAS are dynamic, continuously learning to adapt to external forces, and emerge to new states when necessary to meet unique environmental needs and can't be predicted by the characteristics of the parts.
- Complex Adaptive System require Chefs, not recipe book users, adapting to local context and combining core ingredients in different ways to feed people.
- Context is Key

1. Start from where you are. Map the current dispositional state of the culture and look for patterns.
2. Set a vector (direction to move in). What do we amplify or dampen? Look for adjacent possibles, the first stepping stone.
3. Design interventions (nudges) in order to encourage behavioral change. Run multiple parallel safe-to-fail experiments.
4. Continuous monitoring to determine any shift in direction and landscape changes and develops, plus weak signals.

- Context recognizes that no two situations are identical in a complex adaptive system.
- You cannot apply a case-based approach.

Scrum

Scrum - DasScrumTeam

Scrum is a lightweight framework that helps people, teams, and organizations generate value through adaptive solutions for complex problems. Created by Ken Schwaber and Jeff Sutherland in the early 1990s and first presented at the OOPSLA Conference in 1995, Scrum operates on three foundational pillars: transparency, inspection, and adaptation. Teams work in fixed time periods called Sprints with defined roles (Scrum Master, Product Owner, Developers) and specific events (planning, daily standups, reviews, retrospectives). The framework is guided by five core values: Commitment, Focus, Openness, Respect, and Courage.

Scrum@Scale

Scrum@Scale Guide

Scrum@Scale is a framework for scaling Scrum practices across multiple teams and large organizations, created by Dr. Jeff Sutherland, one of the original Scrum framework developers. The framework addresses scaled roles, scaled events, and organizational structures needed to maintain agile principles while coordinating work across distributed teams. Rather than abandoning Scrum's core tenets at scale, Scrum@Scale applies consistent agile values and practices—including iterative development, regular feedback loops, and self-organizing teams—to enterprise-wide implementations.

Self-Adapting Language Models

Self-Adapting Language Models - arXiv

This paper introduces Self-Adapting Language Models (SEAL), a framework that enables large language models (LLMs) to adapt themselves by generating their own synthetic training data and finetuning directives. The authors—Jyo Pari, Shivam Duggal, Idan Shenfeld, Seungwook Han, Jeremy Bernstein, Akarsh Kumar, Linlu Qiu, Juno Kim, Brian Cheung, Moritz Reuss, Ayush Sekhari, Zhang-Wei Hong, Mehul Damani, Leshem Choshen, and Ryan Yang—propose that LLMs can improve their performance on new tasks and incorporate new knowledge by creating and learning from their own self-edits.

SEAL uses a reinforcement learning loop, where the model generates self-edits (such as restructured information, optimization hyperparameters, or data augmentation instructions) and is rewarded based on improved downstream performance. This approach differs from traditional methods that rely on static data or external adaptation modules. Instead, SEAL allows the model to control its own adaptation process directly.

The framework is tested in two main areas: knowledge incorporation (adding new factual information to the model) and few-shot learning (generalizing from a small number of examples). In both cases, SEAL outperforms standard baselines and even synthetic data generated by

larger models like GPT-4.1. The results show improved accuracy in question answering and better adaptation to new tasks.

The authors also discuss limitations such as catastrophic forgetting (where new learning can overwrite old knowledge), computational overhead, and the need for explicit downstream tasks for evaluation. They suggest that future work could address these challenges and extend SEAL to continual learning and agentic systems.

In summary, SEAL represents a step towards more autonomous and adaptable language models, capable of self-directed learning and improvement. This could be especially important as the availability of human-generated training data becomes limited, making synthetic self-generated data increasingly valuable for future model development. For more details, see the full research paper on SEAL and its implications for future AI development.

Simon Wardley on Evolving Software Engineering Practices with Agentic AI

LinkedIn Post by Simon Wardley

Simon Wardley defines the evolution of software engineering practices with agentic AI as a shift from traditional, deterministic approaches to more dynamic, ecological models. He highlights that agentic AI systems, which operate based on intent and can be unreliable or non-deterministic, open new possibilities for resilience by mimicking the adaptive and robust qualities found in nature. Wardley, referencing the work of C.S. Holling and the classic Dobson & Randell paper, emphasizes that these emerging practices are still developing, but they promise to create highly reliable systems from unreliable components by leveraging ecological principles rather than just engineering ones.

Spotify

Spotify Engineering Culture Part 1 - Henrik Kniberg

Spotify, as described by author Henrik Kniberg, is a company with a unique engineering culture that values agility and continuous improvement. The organization is structured into squads and tribes, each with a specific focus and autonomy over their work. This structure promotes collaboration and innovation, while also allowing for variation and adaptation as the company grows. Spotify's culture encourages trust, learning from failure, and maintaining a healthy work environment to avoid unnecessary bureaucracy.

Sprint

> The Sprint - Scrum Guide

A Sprint, as defined by Ken Schwaber and Jeff Sutherland in the Scrum Guide, is a fixed-length event of one month or less, serving as the heartbeat of Scrum. It is a time-boxed period during which a Scrum Team works to turn ideas into value, ensuring consistency and predictability. Each Sprint includes all necessary work to achieve the Product Goal, such as Sprint Planning, Daily Scrums, Sprint Review, and Sprint Retrospective.

The Sprint is a Empirical Product Control Loop.

> [!info] A Sprint of a Scrum Team in an Arena is compatible with the Match in AME3. But a Match is not necessarily a Sprint.

The Execution Trap

> The Execution Trap - Harvard Business Review

The Execution Trap, as discussed by Roger L. Martin, refers to the misconception that execution is separate from strategy in management. This belief suggests that a mediocre strategy executed well is preferable to a great strategy executed poorly. The idea gained popularity in the early 2000s, with influential figures like Jamie Dimon and Larry Bossidy emphasizing the importance of execution over strategy.

The Lean Startup

The Lean Startup - Wikipedia

The Lean Startup is a methodology for developing businesses and products that aims to shorten product development cycles and rapidly discover if a proposed business model is viable. Created by entrepreneur Eric Ries and published in 2011, it draws from his experiences as a startup advisor and founder, combining ideas from Lean manufacturing and Agile development. The core principles include rapid prototyping, validated learning, iterative product releases, and the concept of the Minimum Viable Product (MVP), enabling entrepreneurs to test hypotheses and adapt strategies through experimentation and continuous customer feedback rather than relying on extensive long-term planning.

The Magical Number Seven Plus or Minus Two

The Magical Number Seven, Plus or Minus Two - Wikipedia

The Magical Number Seven, Plus or Minus Two is a concept introduced by cognitive psychologist George A. Miller, a Harvard psychologist, which suggests that the average person can hold about seven items (plus or minus two) in their short-term memory. This idea highlights the limited number of information "chunks" people can process at one time, and has influenced research and practice in psychology, management, and information design.

The New New Product Development Game

The New New Product Development Game - Harvard Business Review

The article, "The New New Product Development Game," by Hirotaka Takeuchi and Ikujiro Nonaka, explores a revolutionary approach to product development. Instead of the traditional sequential or "relay race" approach to product development, the article suggests a holistic or "rugby" approach where a Team moves as a unit up and down the field. This approach fosters

flexibility, speed, and a more efficient use of resources. The article also highlights the importance of self-organizing Teams, where individuals are given autonomy and encouraged to take initiative. The "rugby" approach to product development is a game-changer, offering a more dynamic and effective method for companies to navigate the complex process of product development.

The Principles of Product Development Flow: Second Generation Lean Product Development

The Principles of Product Development Flow - Google Books

A comprehensive book by Donald G. Reinertsen (2009) that challenges conventional product development management by identifying invisible, unmanaged queues as the root cause of poor development performance. The book presents 175 principles across eight major areas including economic decision-making, queue management, batch size reduction, WIP constraints, accelerated feedback, flow management amid variability, and decentralized control. It applies lean manufacturing concepts and queuing theory specifically to product development, showing how to improve speed, quality, and efficiency.

The SAFe Delusion

The SAFe Delusion

"The SAFe Delusion" is a comprehensive document that provides a critical review of the Scaled Agile Framework (SAFe). It is a curated collection of facts, evidence, and opinions from various sources, intended to assist decision-makers in making informed choices. The document includes case studies, expert opinions, and practitioner experiences, highlighting both the strengths and weaknesses of SAFe.

Wardley Maps

Wardley Maps - On Being Lost

The linked article is the first chapter of Simon Wardley's book, published as a series on Medium. It contains the table of contents for all chapters.

A Wardley Map is a strategic visualization framework created by Simon Wardley. It depicts a value chain, from users and their needs down to the capabilities required to meet those needs. Components are arranged by dependency and mapped against four stages of evolution: Genesis, Custom, Product, and Commodity. Inspired by military strategy principles from Sun Tzu's *Art of War*, Wardley Mapping helps organizations understand who benefits from their work, what those benefits are, and the system of capabilities that deliver market success. This open-source methodology enables leaders at all levels to identify strategic opportunities, avoid costly mistakes, and make better decisions through visual clarity.

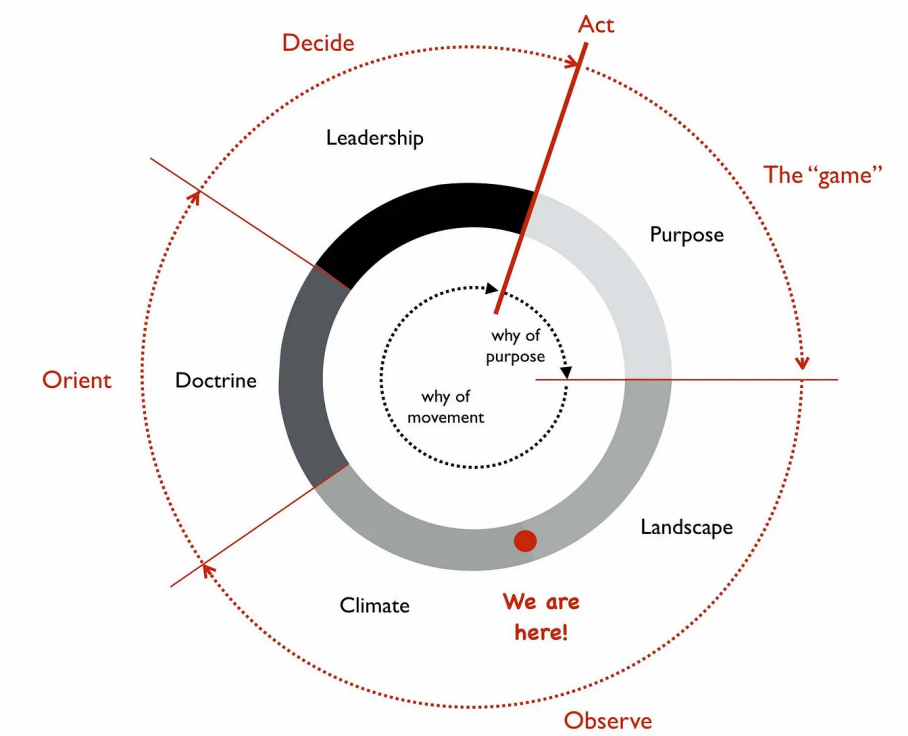
The primary philosophy behind Wardley Mapping is that decision-making improves when we have a clear understanding of:

1. The beneficiaries of our work,
2. The specific benefits they receive, and
3. The system of capabilities that synergistically work to deliver these benefits in the market.

Wardley Mapping Loop Compated with OODA

> Exploring the map - Medium

Wardley's strategy cycle combines John Boyd's OODA loop (Observe, Orient, Decide, Act) with Sun Tzu's five factors of strategy (Purpose, Landscape, Climate, Doctrine, Leadership). Created by Simon Wardley, this framework transforms decision-making from reactive to systematic by emphasizing situational awareness and continuous learning. The cycle addresses the problem of staying stuck in execution mode without reassessing whether solutions address the right problems, using Wardley Maps as visual tools in the Orient phase to connect observations to strategic decisions and actions.



Wardley Mapping

https://medium.com/wardleymaps/on-being-lost-2ef5f05eb1ec

Wardley Mapping is a strategic decision-making process (**leadership**) that hinges on the **purpose** ("the game"), a comprehensive depiction of the competitive **landscape** (illustrated by a map), the external forces influencing the landscape (**climate**), and the development of your team (**doctrine**).

Simon Wardley, in his quest to comprehend strategy evaluation, discovered insights in military history and *Sun Tzu: The Art of War*. For Simon, the five elements outlined by Sun Tzu — Purpose, Landscape, Climate, Doctrine, and Leadership — encapsulated the essential considerations for strategic decision-making.

In a military setting, a map serves as the foundation for understanding the landscape. However, in the absence of a physical competitive landscape in

business, what could serve as its equivalent?

Wardley Maps and Cynefin

> Wardley Maps and Cynefin - Simon Wardley (Medium)
>
> Whilst each system is **complicated** (having many components operating in a defined and known manner), the "right" solution is still emerging. This means the problem space itself is **complex**. Within that one pipeline you can have multiple **complicated** solutions to a **complex** space which is still emerging.

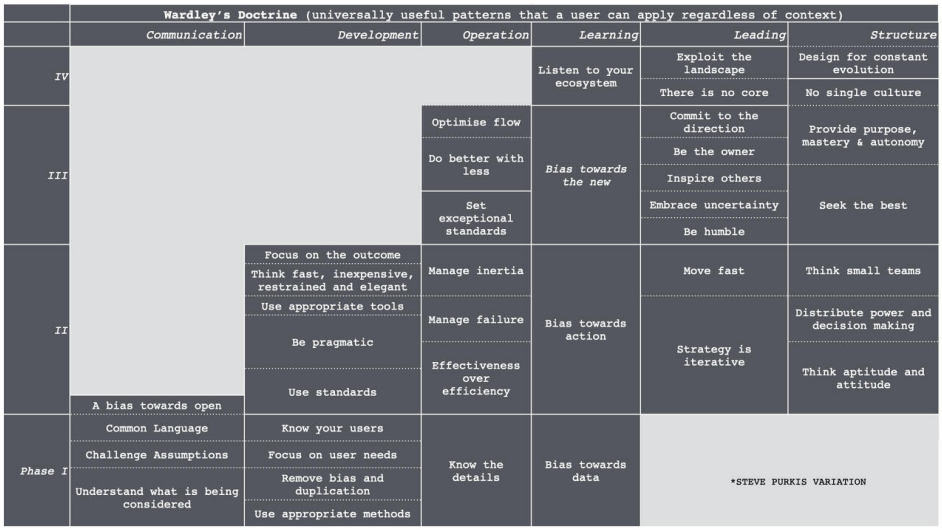
In the article "Wardley Maps and Cynefin" by Simon Wardley, the author discusses the complementary nature of Wardley Maps and the Cynefin framework. Wardley emphasizes that these tools should not be combined into a single holistic view but rather used together to explore problem spaces from different perspectives. He uses a simple map example to illustrate key concepts such as chains of relationships, logical ANDs and ORs, perspectives, and the independent evolution of components within a pipeline.

Wardley explains that while Wardley Maps represent different stages of evolution, Cynefin terms like "complex" and "complicated" cannot be directly mapped onto these stages. Instead, both frameworks should be used in tandem to provide diverse viewpoints. He advises against merging these frameworks into one and highlights the importance of maintaining their distinct perspectives.

Wardley's Doctrines Table

Wardley's Doctrines Table is a visual reference framework that systematizes Simon Wardley's universally applicable principles for organizational decision-making and strategic planning. The table organizes doctrines into categories such as communication, development, operation, learning, leading, and structure, providing a comprehensive checklist for assessing organizational maturity. Originally developed as part of Wardley Mapping methodology, these doctrines serve as context-independent guidelines

that help organizations understand their landscape, anticipate change, and adapt strategies accordingly.



Sources and Definitions

This section serves as the reference glossary for AME3. Here you'll find definitions of key concepts, methodologies, and terminology used throughout the framework, along with links to the authoritative sources that inform our approach.

eXtreme Programming

> What is Extreme Programming? - Ron Jeffries

eXtreme Programming (XP), as described by Ronald E. Jeffries, Kent Beck and others, is a discipline of software development that emphasizes values such as simplicity, communication, feedback, courage, and respect. It involves the entire team working together using simple practices and receiving enough feedback to understand their position and adjust their practices to their unique situation. Key elements of eXtreme Programming include pair programming, test-driven development, continuous integration, and maintaining a sustainable pace. The team is responsible for all code, ensuring

it is written in a consistent pattern, and the system is kept running and integrated at all times.

> Extreme Programming - Martin Fowler

Extreme Programming (XP), as described by Martin Fowler, is a software development methodology developed primarily by Kent Beck. It is one of the first agile methods and was dominant in the late 90s and early 00s. XP emphasizes a progression from broad values to concrete practices, combining technical and management practices. It popularized practices like continuous integration, refactoring, Test-Driven Development, and agile planning.

It's Always a People Problem

> "No matter how it looks at first, it's always a people problem."
>
> — Gerald Weinberg

A famous quote from Gerald Weinberg, emphasizing that technical problems in software development and systems thinking are fundamentally about people, communication, and human factors rather than purely technical issues. This insight highlights the importance of addressing interpersonal dynamics, team collaboration, and organizational culture when solving complex problems.

Pair Programming

> Pair programming - Wikipedia

Pair programming is a software development technique where two programmers work together at one workstation. One acts as the "driver," writing code, while the other, the "observer" or "navigator," reviews each line and considers the broader direction. This method, described by authors such as Laurie Williams and Robert Kessler, aims to improve code quality, enhance learning, and foster better communication within the development team.

No plan survives first contact with the enemy

> *"No plan of operations extends with any certainty beyond the first encounter with the main enemy forces."*

Helmuth von Moltke, Prussian field marshal and Chief of the Prussian General Staff, wrote this in 1871 reflecting on military strategy. This observation about the limits of planning in the face of uncertain contact with adversaries evolved into the widely cited modern version: *"No plan survives first contact with the enemy."* Von Moltke's insight predates modern agile methodology by over a century, yet captures the same fundamental truth: in complex, uncertain environments, rigid adherence to predetermined plans fails because reality diverges from expectations once execution begins.

Helmuth von Moltke, "Über Strategie" (1871), in: Militärische Werke, Band 2, Teil 2, Mittler & Sohn Berlin 1900, S. 291 | Quote Investigator Analysis

Overconfidence Bias

> PMC - Overconfidence over the lifespan | Wikipedia - Overconfidence Effect | Scribbr - Overconfidence Bias

Overconfidence Bias is a cognitive bias where people systematically have more confidence in the accuracy of their knowledge and judgments than is objectively justified. Research originated in psychological studies during the 1970s-1980s, with significant contributions from researchers like Baruch Fischhoff and Paul Slovic. The bias manifests in three distinct forms: overestimation (overrating actual performance), overplacement (overrating performance relative to others), and overprecision (excessive certainty in the accuracy of beliefs). Classic studies using 90% confidence intervals consistently show that people's actual accuracy rates fall around 50%, demonstrating a dramatic miscalibration between perceived and actual competence. Social psychologist Scott Plous called it the most prevalent and potentially catastrophic problem in judgment and decision-making.

Planning Fallacy

> Kahneman & Tversky (1994) - Planning Fallacy Study | Wikipedia - Planning Fallacy

Planning Fallacy describes the tendency to underestimate time, costs, and risks of future actions while simultaneously overestimating their benefits. Research by Kahneman and Tversky (1979) and Buehler demonstrated this bias empirically: In a study of 37 psychology students estimating thesis completion time, only 13% finished by their 50% probability estimate, only 19% by their 75% estimate, and only 45% by their 99% estimate. The bias stems from wishful thinking, self-serving attribution of delays to external factors, and taking an "inside view" focused on specific task details rather than historical data from similar past projects.

Plans are worthless, but planning is everything

> *"Plans are worthless, but planning is everything."* - Dwight D. Eisenhower

Dwight D. Eisenhower, 34th President of the United States, stated this at the National Defense Executive Reserve Conference in 1957. The full context reveals deeper meaning: *"In preparing for battle I have always found that plans are useless, but planning is indispensable."* Eisenhower explained that emergencies are by definition unexpected, so they won't unfold as planned. Yet the planning process itself creates understanding, preparedness, and adaptability. This insight, drawn from his military experience as Supreme Commander of Allied Forces in World War II, became foundational to modern agile thinking about iterative planning over rigid adherence to fixed plans.

Remarks at the National Defense Executive Reserve Conference (1957) | Quote Investigator Analysis

A Small Matter of Programming

> A Small Matter of Programming - MIT Press

"A Small Matter of Programming: Perspectives on End User Computing" is a book by Bonnie Nardi, published by MIT Press in 1993. Based on anthropological fieldwork, Nardi studied how end users were already building their own tools using spreadsheets, CAD systems, and other customizable software, far beyond what vendors intended. She documented that non-programmers routinely created sophisticated applications to solve domain-specific problems, challenging the assumption that software development belongs exclusively to professional programmers. The book is an early and influential articulation of the gap between what packaged software delivers and what users actually need, a theme that resurfaces in the Malleable Software[9] movement and in AI-driven, situation-oriented computing.

All models are wrong, but some are useful

"All models are wrong, but some are useful." - George E. P. Box

George E. P. Box (1919–2013), britischer Statistiker, formulierte diesen Grundsatz erstmals in seinem Aufsatz *Science and Statistics* (1976) im Journal of the American Statistical Association. Die vollständige Passage lautet: *"Since all models are wrong the scientist cannot obtain a 'correct' one by excessive elaboration. On the contrary following William of Occam he should seek an economical description of natural phenomena."* Box wiederholte und erweiterte den Gedanken 1987 in *Empirical Model-Building and Response Surfaces*: *"All models are wrong, but some are useful."* Das Zitat wurde zu einem Leitsatz in Wissenschaft, Statistik und agilem Denken – es erinnert daran, dass Modelle Vereinfachungen der Realität sind und ihr Wert nicht in ihrer Korrektheit liegt, sondern in ihrer Nützlichkeit für Entscheidungen und Verständnis.

Wikipedia - All models are wrong | Box (1976) - Science and Statistics

Ambidextrous Organizations: Managing Evolutionary and Revolutionary Change

Tushman, M.L. & O'Reilly, C.A. (1996) - California Management Review, Vol. 38, No. 4, pp. 8-30

[9] see: Malleable Software, 227

Award-winning paper by Michael Tushman and Charles O'Reilly III, winner of the Andersen Consulting Award for most impactful article. Organizations evolve through periods of incremental (evolutionary) change punctuated by discontinuous (revolutionary) change. The challenge for managers is to simultaneously pursue both types of innovation. The authors introduce the concept of structural Organizational Ambidexterity[10]: creating separate units with distinct structures, cultures, and processes for exploitation and exploration, integrated at the senior leadership level.

Competing Against Luck

Competing Against Luck - HarperCollins (2016)

Competing Against Luck: The Story of Innovation and Customer Choice, published in 2016 by Clayton M. Christensen, Taddy Hall, Karen Dillon, and David S. Duncan. The book formalizes the Jobs to be Done[11] theory: customers don't buy products, they "hire" them to fulfill a specific job in their lives. Innovation succeeds when it addresses these jobs better than existing alternatives. Christensen argues that understanding the job, not the customer demographics, is the causal mechanism behind successful innovation. The framework shifts focus from "who is the customer" to "what progress is the customer trying to make," providing a practical lens for product strategy and competitive differentiation.

Computer Lib / Dream Machines

Computer Lib / Dream Machines (Wikipedia)

Self-published in 1974 by Ted Nelson, Computer Lib / Dream Machines is a two-sided manifesto challenging the "priesthood of computing," the idea that only specialists should control and understand computers. Nelson argued that computers are universal creative media, not tools reserved for experts. The book introduced early concepts of hypertext and interactive, non-linear media, years before the World Wide Web made those ideas mainstream. Written in a deliberately accessible, zine-like style, it called

[10] see: Organizational Ambidexterity: Past, Present and Future, 228
[11] see: Jobs to be Done, 225

on ordinary people to seize computing for themselves. It became a foundational text for the personal computing movement and continues to shape thinking about open, user-empowering software.

If I had asked people what they wanted, they would have said faster horses

> *"If I had asked people what they wanted, they would have said faster horses."* - attributed to Henry Ford

This quote is widely attributed to Henry Ford (1863–1947), founder of the Ford Motor Company and pioneer of mass production. However, no written record from Ford himself has been found. Quote Investigator traces the earliest known version to a 2002 article by John McNeece in the Cruise Industry News Quarterly. Despite its disputed origin, the quote became a cornerstone in product development and innovation thinking. It illustrates a fundamental insight: customers can articulate problems within their current frame of reference, but rarely envision transformative solutions. In complexity terms, the quote highlights the limits of agreement on the **What** axis — what people say they want may not reflect what they actually need, especially when the problem space is genuinely uncertain or novel.

Quote Investigator — Faster Horse | Harvard Business Review — Henry Ford, Innovation, and That "Faster Horse" Quote

Jobs to be Done

> Competing Against Luck - Christensen et al. (2016) | The Innovator's Solution - Christensen & Raynor (2003)

Jobs to be Done (JTBD) is an innovation framework stating that customers do not buy products or services, they "hire" them to make progress in a specific circumstance. Originated by Clayton M. Christensen at Harvard Business School, first introduced in *The Innovator's Solution* (2003) with Michael Raynor, then formalized in Competing Against Luck[12] (2016) with

[12] see: Competing Against Luck, 224

Taddy Hall, Karen Dillon, and David S. Duncan. The core insight: understanding the "job" the customer is trying to get done, not demographics or product attributes, is the causal mechanism behind successful innovation. The framework shifts product strategy from "who is the customer" to "what progress is the customer trying to make in this situation." Theodore Levitt's famous dictum captures the spirit: "People don't want to buy a quarter-inch drill. They want a quarter-inch hole."

Kano Model

> Attractive Quality and Must-Be Quality (1984)

The Kano Model is a theory of product development and customer satisfaction developed by Professor Noriaki Kano and colleagues (Seraku, Takahashi, Tsuji) at Tokyo University of Science, first published in 1984 in the Journal of the Japanese Society for Quality Control. The model classifies product and service attributes into five Quality Attributes (, hinshitsu yōso): Must-be Quality ( ), One-dimensional Quality ( ), Attractive Quality ( ), Indifferent Quality ( ), and Reverse Quality ( ). A key insight is that quality attributes migrate over time: today's Attractive Quality becomes tomorrow's Must-be Quality as customer expectations evolve.

Large Language Model

> Wikipedia - Large language model | IBM - What Are Large Language Models

A Large Language Model (LLM) is a neural network with billions of parameters, trained on vast amounts of unlabeled text using self-supervised learning. LLMs generate, summarize, translate, and reason over natural language. The foundation was the Transformer architecture, introduced by Vaswani et al. in the landmark paper "Attention Is All You Need" (NeurIPS, 2017). OpenAI's GPT (2018) and Google's BERT (2018) were the first widely recognized LLMs. Subsequent models like GPT-4, Claude, Gemini,

and DeepSeek scaled to hundreds of billions of parameters, enabling capabilities from code generation to complex reasoning. LLMs are the core technology behind Generative Artificial Intelligence[13].

Malleable Software

> Designing and Programming Malleable Software - Tchernavskij (2019) | Malleable Software in the Age of LLMs - Litt (2023) | Malleable Software - Ink & Switch (2025) | Webstrates - Klokmose et al. (2015)

Malleable Software is the concept that users should be able to reshape and modify their software tools at runtime, rather than waiting for vendor updates. The term was coined by Philip Tchernavskij in his 2019 doctoral thesis at Universite Paris-Saclay. He identifies that traditional packaged applications lock users out of meaningful modification and proposes design patterns ("entanglers") for reusable, composable UI components. Geoffrey Litt (Ink & Switch) extended the concept in 2023, arguing that LLMs are the missing enabling technology: non-programmers can now specify intent in natural language and create personal tools on the fly. The 2025 Ink & Switch manifesto by Litt, Horowitz, van Hardenberg and Matthews synthesizes the vision: local-first computing combined with AI restores user agency in a world of locked-down apps.

OpenClaw

> OpenClaw | Serenities AI Deep Dive | TechTarget Explainer

OpenClaw is a free, open-source autonomous AI assistant developed by Peter Steinberger, released November 2025. Unlike conversational AI tools (ChatGPT, Claude), OpenClaw runs as a persistent local daemon that initiates actions independently via its Heartbeat system (cron-scheduled autonomous tasks). Users interact through messaging platforms (WhatsApp, Telegram, Slack, Signal) rather than a dedicated app. The agent maintains persistent memory in plain Markdown files and supports 50+ integrations. With 234K GitHub stars and 300-400K users by early 2026, it represents a

[13] see: Generative Artificial Intelligence, 197

shift from conversation-as-interface to agent-as-infrastructure. In February 2026, Steinberger announced joining OpenAI, with the project moving to an open-source foundation.

Organizational Ambidexterity: Past, Present and Future

> O'Reilly, C.A. & Tushman, M.L. (2013) - Academy of Management Perspectives, Vol. 27, No. 4, pp. 324-338

Comprehensive literature review by Charles O'Reilly III and Michael Tushman surveying 15 years of research on Organizational Ambidexterity. The authors define ambidexterity as the ability to compete in mature technologies and markets (where efficiency, control, and incremental improvement are prized) while also competing in new technologies and markets (where flexibility, autonomy, and experimentation are needed). The paper reviews structural, contextual, and sequential forms, highlights what is known and unknown, and identifies promising areas for ongoing research.

Personal Dynamic Media

> Personal Dynamic Media (PDF)

Published in 1977 by Alan Kay and Adele Goldberg in IEEE Computer, this paper describes the Dynabook concept and the Smalltalk programming environment developed at Xerox PARC. Its central idea is "symmetric authoring and consuming": every user should be able to shape their own tools, not just use pre-built applications. Kay and Goldberg envisioned a portable personal computer as a dynamic medium for creative thought, learning, and simulation. The paper demonstrated children and adults creating their own programs in Smalltalk. It laid the intellectual foundation for modern laptops, tablets, and the broader vision of end-user programming that still drives software innovation today.

Proactive Computing

> Proactive Computing (Semantic Scholar)

David Tennenhouse's 2000 article in Communications of the ACM coined the term "proactive computing." Tennenhouse, then director of research at Intel, argued that computing must shift from reactive to proactive: instead of waiting for human commands, systems should anticipate needs and act autonomously. The central idea is that humans leave the interaction loop entirely, and the machine observes, decides, and acts on their behalf. This paper is the most direct academic articulation of the principle that underlies autonomous agent systems and situation-oriented software. Where Weiser made computing invisible, Tennenhouse made it self-directed.

SaaS Will Collapse in the Agent Era

> Satya Nadella on the Future of AI Agents and the End of SaaS (Forward Future)

Satya Nadella, CEO of Microsoft, argued in interviews during 2024 and 2025 that traditional SaaS will lose its foundation as AI agents mature. His core claim: most business applications are fundamentally CRUD databases with a logic layer on top. AI agents will absorb that logic layer, orchestrating workflows directly against the underlying data. What remains is data and governance, not packaged software. This perspective reflects Microsoft's strategic pivot toward Copilot and agent-based computing, where intelligence lives in the agent, not in the application.

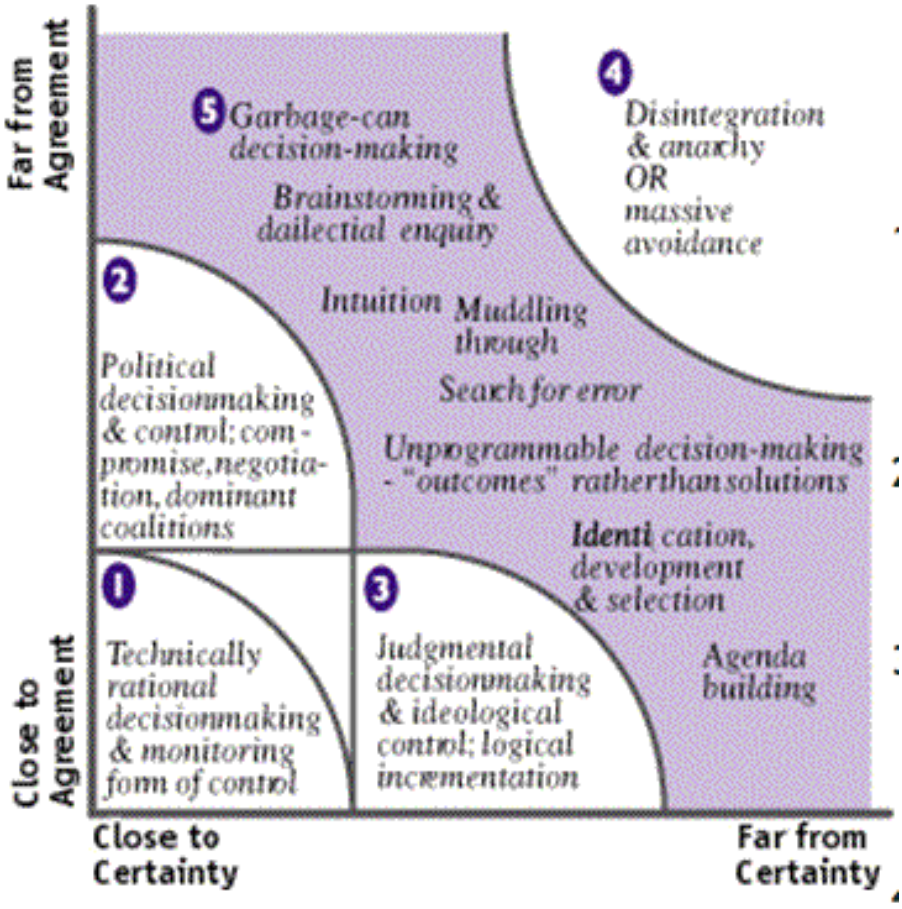
Stacey Matrix

> Stacey, R.D. (1996). *Strategic Management and Organisational Dynamics.* 2nd ed. Prentice Hall | Habermann, F. (2020). Agile Populism — The Stacey Matrix

The Stacey Matrix (also known as the Agreement & Certainty Matrix) was introduced in 1996 by Ralph D. Stacey in the 2nd edition of his textbook. The original diagram uses two axes — **Agreement** and **Certainty** — and

distinguishes five decision-making zones: (1) technically rational, (2) political, (3) judgmental, (4) zone of complexity, and (5) chaos/anarchy.

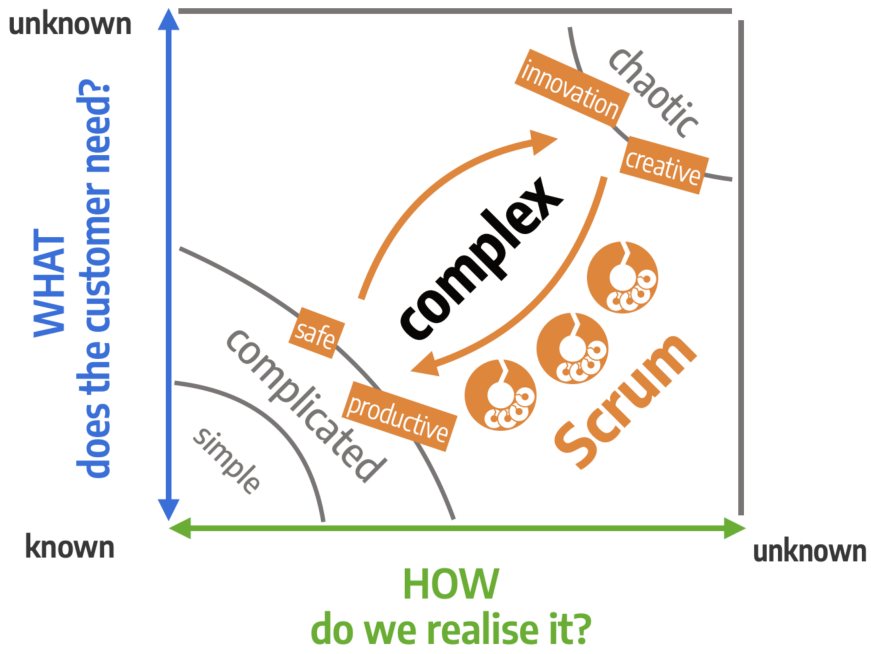
Stacey himself later regretted the diagram and removed it from the 4th edition (2003): *"I no longer use the diagram because it is simply interpreted in a way that sustains the dominant discourse while using the alternative jargon of complexity."*



Ralph D. Stacey's original matrix

The widely used version with the zones "Simple, Complicated, Complex, Chaotic" does not originate from Stacey but from Brenda Zimmerman (~2001), further modified by Ken Schwaber (2004), who renamed the axes to "Requirements" and "Technology" without justification.

Peter Beck has renamed the axes to **What** and **How** and uses the model extensively in Scrum training exercises. Contrary to Stacey's own critique, he finds the model highly valuable in discussions with participants. It exposes the organizational silos commonly found in enterprises: Business/Customer (**What**) vs. IT/Tech/R&D (**How**). The model challenges the widespread belief that "the other side" is simply not performing well enough. It helps participants understand that complexity is real for everyone — including the other side.



Complex Space and Scrum Matrix by Peter Beck

Consider the **What** axis: when an engineer asks a customer what problem they want solved, the answer may be vague, wrong, or absent — not because the customer is incompetent, but because the problem space is genuinely uncertain. As Henry Ford reportedly put it: *"If I had asked people what they wanted, they would have said faster horses."* Sometimes we cannot even ask the right question, because we are dealing with unknown unknowns.

The **How** axis works the same way: when a business stakeholder asks an engineering team to estimate a technically novel project, the honest answer is often "we don't know yet." An engineer cannot give an accurate timeline for a problem they have never solved before — not because they

lack skill, but because the solution space is genuinely complex. Demanding a precise estimate under these conditions does not reduce uncertainty; it merely creates an illusion of certainty.

The 2028 Global Intelligence Crisis

The 2028 Global Intelligence Crisis - Citrini Research

A speculative but well-argued scenario by Citrini Research and Alap Shah exploring how continued AI advancement could trigger an economy-wide crisis through labor displacement and financial instability. The authors argue that AI productivity gains create a destructive feedback loop: companies replace workers to protect margins, displaced workers spend less, consumer demand weakens, and the cycle accelerates. The scenario identifies three interconnected crises: agentic AI disrupting white-collar work, AI agents collapsing intermediary profit models, and income impairment destabilizing the mortgage market. By June 2028, the scenario projects double-digit unemployment, a 38% equity market decline, and rising mortgage delinquencies among prime borrowers.

The Ambidextrous Organization: Designing Dual Structures for Innovation

Duncan, R.B. (1976) - The Management of Organization Design

Foundational book chapter by Robert B. Duncan, published in "The Management of Organization Design: Strategies and Implementation" (Kilman, Pondy, Slevin, eds., North Holland, New York, pp. 167-189). Duncan coined the term "ambidextrous organization," arguing that firms need dual structures to manage innovation: organic, flexible structures for initiation and mechanistic, efficient structures for implementation. This work laid the conceptual groundwork for all subsequent research on Organizational Ambidexterity, influencing March (1991), O'Reilly, Tushman, and others.

The Ambidextrous Organization (HBR 2004)

O'Reilly, C.A. & Tushman, M.L. (2004) - Harvard Business Review, Vol. 82, No. 4, pp. 74-81

Practitioner article by Charles O'Reilly III and Michael Tushman presenting empirical evidence for structural Organizational Ambidexterity. Examining 35 attempts at breakthrough innovation, they found that over 90% of ambidextrous structures succeeded, while none of the cross-functional or unsupported teams reached their goals. Case studies of USA Today and Ciba Vision illustrate how separating exploratory units from traditional ones, with different processes, structures, and cultures, while maintaining tight integration at senior executive level, enables both radical innovation and incremental improvement.

The Computer for the 21st Century

The Computer for the 21st Century (Scientific American)

Mark Weiser's 1991 article in Scientific American is the founding document of ubiquitous computing. Its most famous line: "The most profound technologies are those that disappear." Weiser, chief technologist at Xerox PARC, argued that computing should be woven into the fabric of everyday life, invisible and context-aware, rather than demanding human attention. He predicted a world of embedded sensors, networked devices, and displays at three scales (tabs, pads, boards) that fade into the background. This is the earliest articulation of what we now call situation-oriented computing: machines that fit the human environment instead of forcing humans to enter theirs.

The State of the Art in End-User Software Engineering

The State of the Art in End-User Software Engineering (ACM)

Ko et al., 2011, published in ACM Computing Surveys. This landmark survey synthesizes decades of research on end-user software engineering. Its key finding: far more end-user programmers exist than professional software developers. Spreadsheet users, scientists writing scripts, and web designers all build software without formal training. The paper documents a massive, persistent gap between what people need to build and what professional development can deliver. It remains a foundational reference for the malleable software and no-code movements, framing the challenge of enabling non-programmers to create reliable tools.