LeadingAgile

AGILE ADOPTION GOVERNANCE MODEL

LEADINGAGILE

PREMISE

Organizations must adapt their Agile adoption governance models as the agile adoption progresses to take advantage of their improved ability to execute.

KEY CHALLENGES

- Agile Governance frameworks are designed and optimized to solve a specific, static problem and are not designed to adapt substantially when improvements made during the agile adoption journey in managing business, organizational and systemic dependencies.
    a. SAFe is more successful when deployed to solve for a higher level of organizational and systemic dependencies.
    b. Scrum and Scrum XP are more successful when deployed to solve for a low level of organizational and systemic dependencies.
    c. LeSS is more successful when deployed to solve for a lower level of organizational and systemic dependencies.
    d. Lean Startup is more successful when deployed to solve for a low level of organizational and systemic dependencies.
- Large organizations with significant legacy software investments that are embarking on an Agile Transformation usually begin with a high level of organizational and systemic dependencies and are incented to progress to a lower level of organizational and systemic dependencies in order to improve Time to Value and Quality.

RECOMMENDATIONS

- Establish business goals and define the systemic changes necessary to achieve the business goals.
- Provide an Agile adoption governance model that recognizes the transition from high to low organizational and systemic dependencies and adjusts the framework as organizations achieve higher levels of proficiency.
- Establish a roadmap for organizational and systemic changes necessary to enable the progression. This roadmap begins with making the organization predictable in the delivery of business value and progresses by applying lean techniques, automating test and automated deployment methods to decouple systemic dependencies and by doing so remove many organization dependencies. The organization governance model can then be adjusted to take advantage of the reduction in systemic dependencies.
- Enable an organization to designate the goal state and manage the pace of change, based on business goals and business value over time.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

Organization structures may have many attributes including divisions, profit & loss demarcations, geography, products, and markets, market segments, and customer segments.

The teams supporting these organizational attributes can grouped by functions (sales, marketing, product management, technology, operations), by process (customer, transaction, billing, payment, general & administrative) or matrixed across function and process.

Organizations that have aligned by function have typically focused on optimizing cost. Organizations that have aligned by process have typically valued speed to change higher than cost optimization.

Conway's Law states "Organizations which design systems … are constrained to produce designs which are copies of the communication structures of these organizations" M. Conway, 1968.

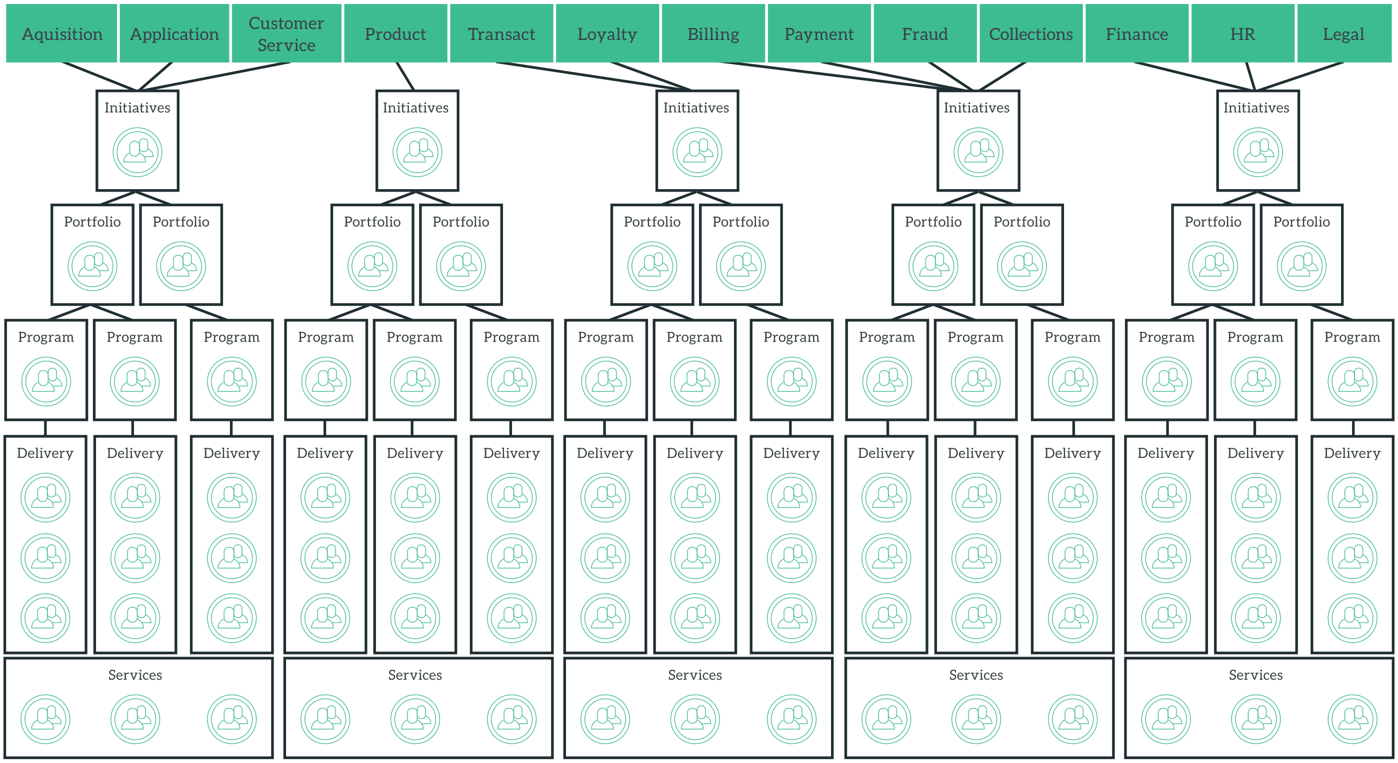
Agile transformation provides an opportunity to realign and optimize the participating organizations, usually Product Management and Technology. Ask these questions to understand the degree that current organization alignment enables Time to Value:

- What business goals are defined for the Value Streams or Customer Journeys?
- What business capabilities in the Value Streams or Customer Journeys enable those goals?
- How are these business capabilities scaled in your organization, and is it commensurate with business value provided?

The realignment of organizational initiatives into Portfolios that are managed as products and services with supporting delivery teams is an important first step for establishing an agile adoption governance model and aligning for "Speed to Change".

Figure 1 depicts an example of a business Value Stream and the organization that has evolved to support it. The organization consists of a strategy tier that defines strategy and business initiatives, a Portfolio tier that decomposes the business initiatives into executable projects or epics, a Program / Product tier that further defines the necessary changes to implement the proposed changes and Delivery and Services teams that develop and test these changes.

Figure 1: Business Capabilities and Organization Structure example



ANALYSIS

The LeadingAgile agile adoption governance model uses an Expedition metaphor and defines five Base Camps:

BASE CAMP 1 - PREDICTABILITY

BASE CAMP 2 - IMPROVE FLOW

BASE CAMP 3 - DECOUPLING

BASE CAMP 4 - AUTONOMOUS

BASE CAMP 5 - INVEST TO LEARN

Figure 2 provides the framework to define the Base Camp progression of individual teams.

Teams are cohesive groups of cross functional members, that work together on the same software product(s) for long periods of time and have demonstrated the ability to predictably produce working, tested software on a prescribed frequency, usually two or three weeks.

Teams that have accomplished **Base Camp 1** have these attributes, behaviors and capabilities: teams are stable, well defined backlogs exist, deliver frequently, measurable progress, and transparency.

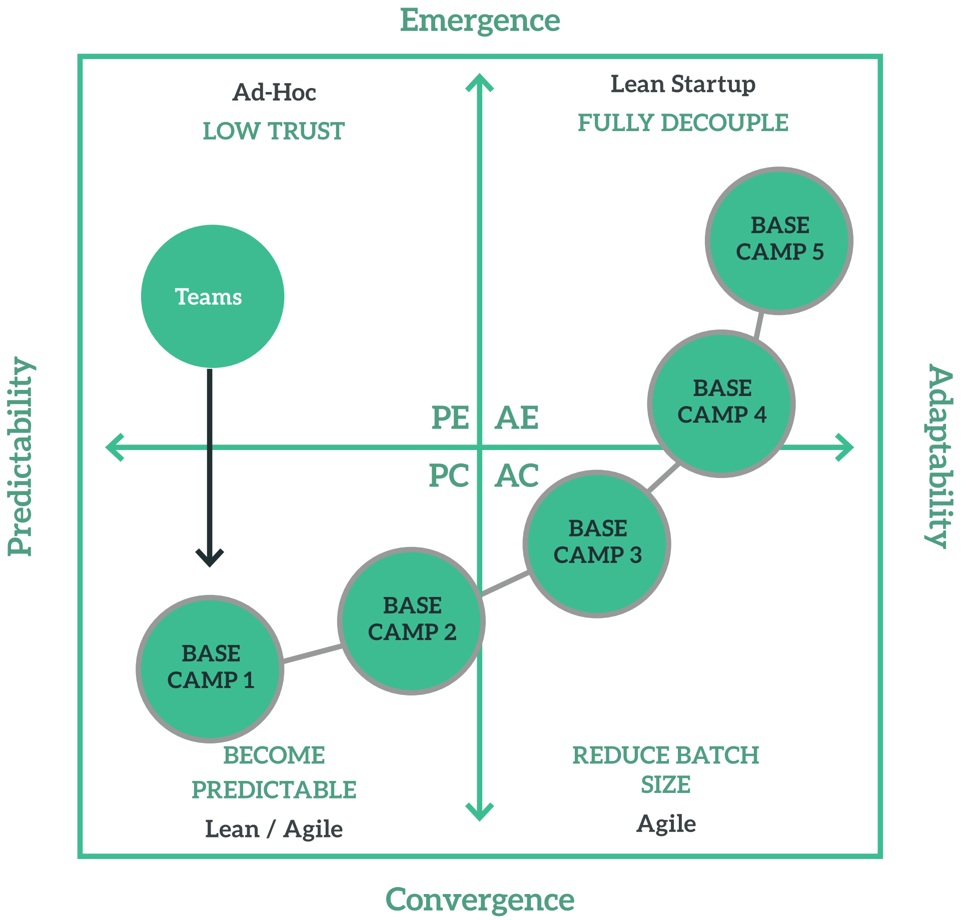
Teams that have accomplished **Base Camp 2** have these attributes, behaviors and capabilities: quarterly agile planning, small batches (releases) at higher frequency, improved coordination between teams, and have introduced flow based metrics.

Teams that have accomplished **Base Camp 3** have these attributes, behaviors and capabilities: Continuous Integration, refactoring of legacy software, release cadence, and DevOps.

Teams that have accomplished **Base Camp 4** have these attributes, behaviors and capabilities: adaptive governance, funding capabilities, economic prioritization, and are Continuous Delivery capable, but business dependencies may require a release cadence.

Teams that have accomplished **Base Camp 5** - Invest to Learn have these attributes, behaviors and capabilities: outcome based accountability, innovation focused, design thinking, Lean Startup, and Continuous Delivery

Figure 2: LeadingAgile – Agile Adoption Governance Model



The **Base Camp 1-2-3** progression requires "orchestrating dependencies" by organizing teams to optimize the management of existing software and organizational dependencies.

The **Base Camp 1-2** progression can be accomplished without a significant investment in Quality Assurance automation, software refactoring and DevOps.

Achieving **Base Camp 2-3** may be sufficient for organizations that believe Continuous Delivery is not necessary to support their business model and customer base.

The **Base Camp 3-4** progression is focused on "encapsulating or eliminating dependencies" by automating software engineering practices for software build, testing, environment build, deployment, Continuous Integration, and Continuous Delivery capabilities and by decoupling monolithic services.

The **Base Camp 2-3-4** progression does require an organization to invest in Quality Assurance automation, software refactoring and DevOps.

When sufficient progress has been achieved on the **Base Camp 3-4** progression the amount of effort required to manage systemic dependencies is significantly reduced. The organization can be (re)organized to focus resources on optimizing continuous flow and lead time to business value.

The Teams at Base Camp 5 are usually built to be adaptive and start and stay at Base Camp 5.

Figure 3 provides an example of Business Themes and corresponding Business Goals. The teams that achieve the Base Camp 3-4 progression will enable the Business Goals in their respective domains.

Figure 3: Base Camp Business Themes and Business Goals

Base Camp	Business Theme	Business Goals
1-2-3	Increase Predictability, Improve Time to Value	Working, tested software released to production every 2 weeks
1-2-3	Improve Quality	Retire accrued technical debt
2-3-4	Improve Quality	No system outages during production release implementations
3-4-5	Improve Lead Time to Value	Product validation with A-B user testing in production environments
3-4-5	Improve Lead Time to Value	Non-Prod environments on demand
3-4-5	Improve Lead Time to Value	Align change control procedures with automated continuous integration and continuous delivery

The Gartner Bimodal approach distinguishes two separate methods of operation. Mode 1 is focused on predictability and convergent business demand, Mode 2 is focused on business hypothesis, rapid experimentation and learning cycles.

The investment in Quality Automation and Continuous Delivery capabilities is required to operate in a Gartner Mode 2 environment. These same investments can be made to Mode 1 business capabilities to achieve many of the benefits of Mode 2 operations.

Figure 4: Agile Adoption Model with Relationship to Gartner Bimodal

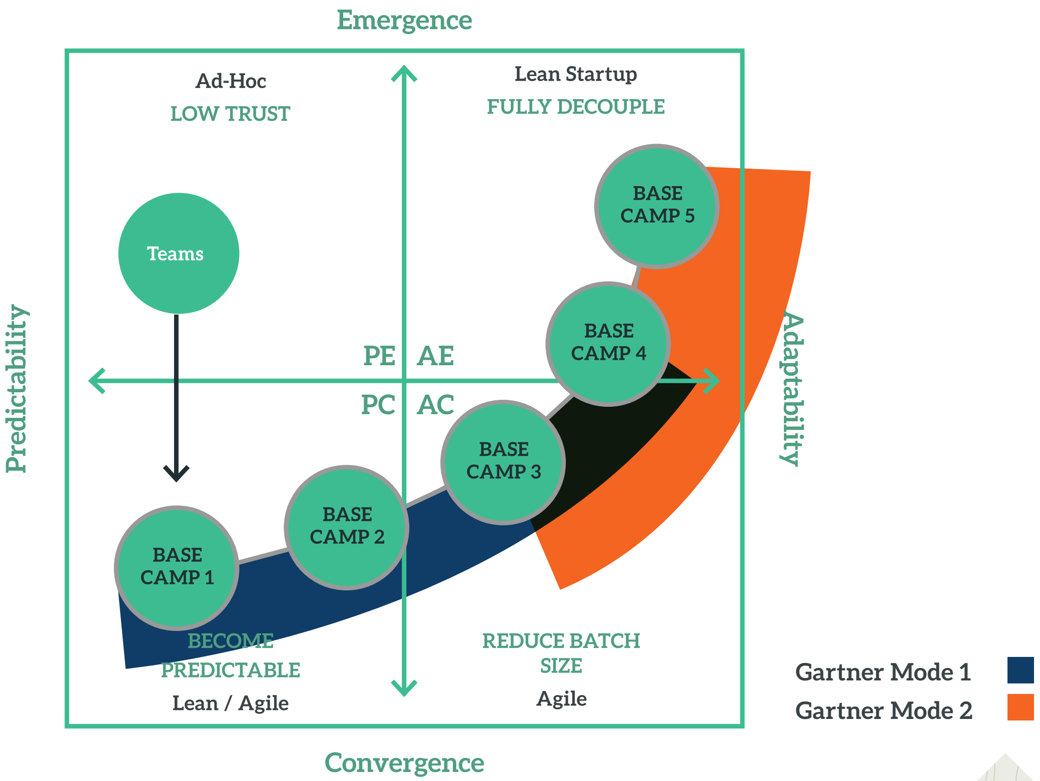


Figure 5 denotes the optimal ranged for economic batch size.

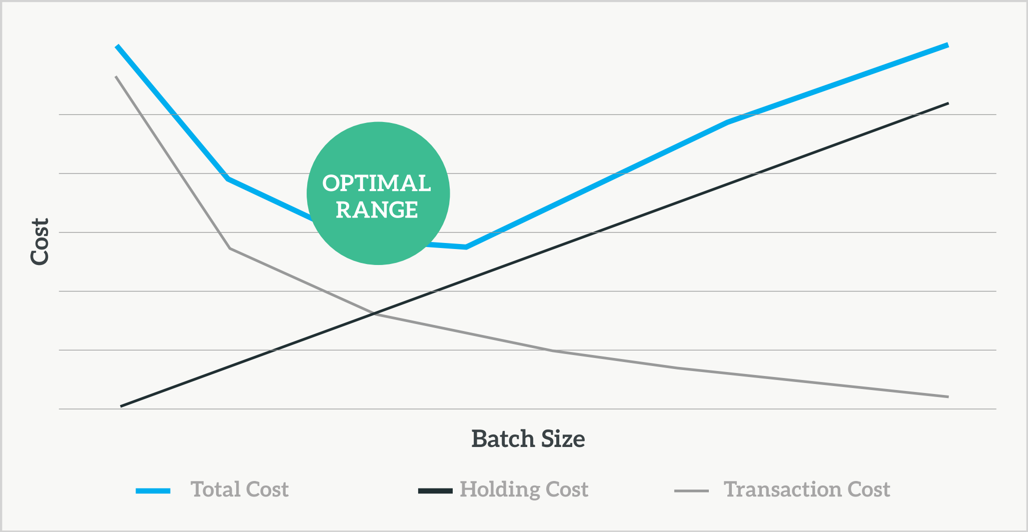
The Economic Batch Size optimizes the quantity, duration and size of batches:

- Holding Costs represent the coordination of change not yet Released
- Longer Release schedules increase Holding Costs
- Transaction costs are the costs of a Release (definition, development, quality, delivery).

The impediments to Time to Value for teams in Base Camp the 1-2-3 progression are Large Batch Sizes and Speed to Change. These are manifested in longer release cycles and large release content. The objective of Base Camp 3-4 progression is to reduce the duration of release cycles. This has the effect of delivering Value on a higher frequency. Optimizing the Economic Batch Size for Product Releases will improve Time to Value.

The high transaction costs associated with small batch sizes is indicative of a large fixed cost relative to a small batch size (release). This is the next target for optimization.

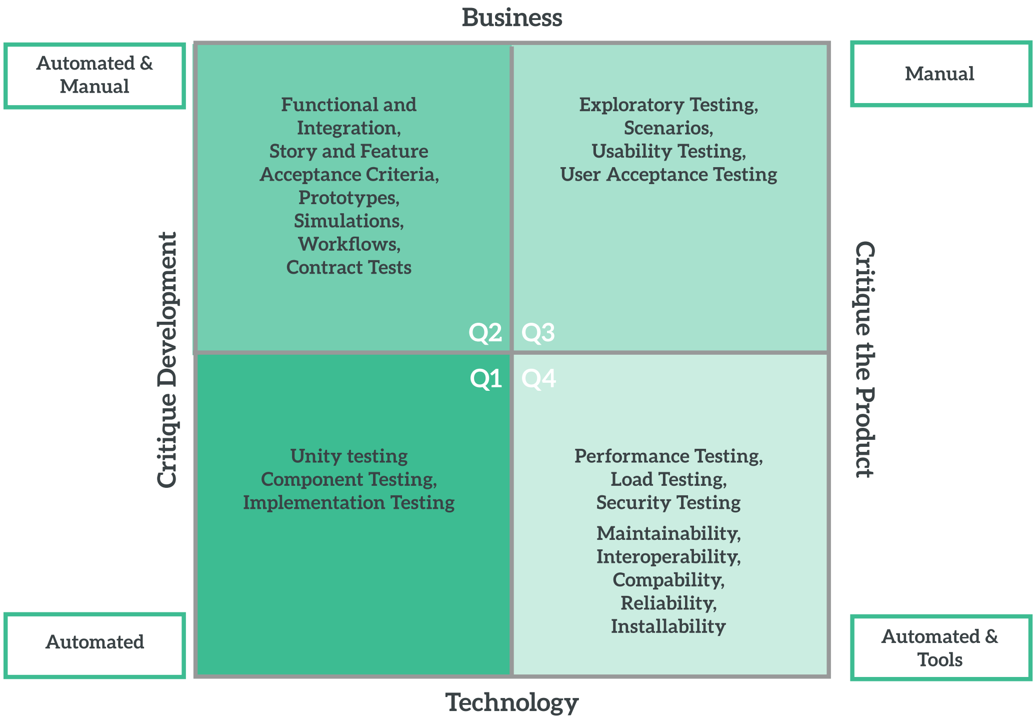
Figure 5: Economic Batch Size



Reinertsen, Principles of Product Development Flow, 2009

Figure 6 indicates the focus on Quality Automation that is required to reduce the fixed cost of a Release.

Figure 6: Batch Size Optimization Methods – Testing



Business

Automated & Manual

Functional and Integration, Story and Feature Acceptance Criteria, Prototypes, Simulations, Workflows, Contract Tests

Q2

Q3

Exploratory Testing, Scenarios, Usability Testing, User Acceptance Testing

Manual

Critique Development

Critique the Product

Q1

Q4

Unity testing Component Testing, Implementation Testing

Performance Testing, Load Testing, Security Testing

Maintainability, Interoperability, Compability, Reliability, Installability

Automated

Automated & Tools

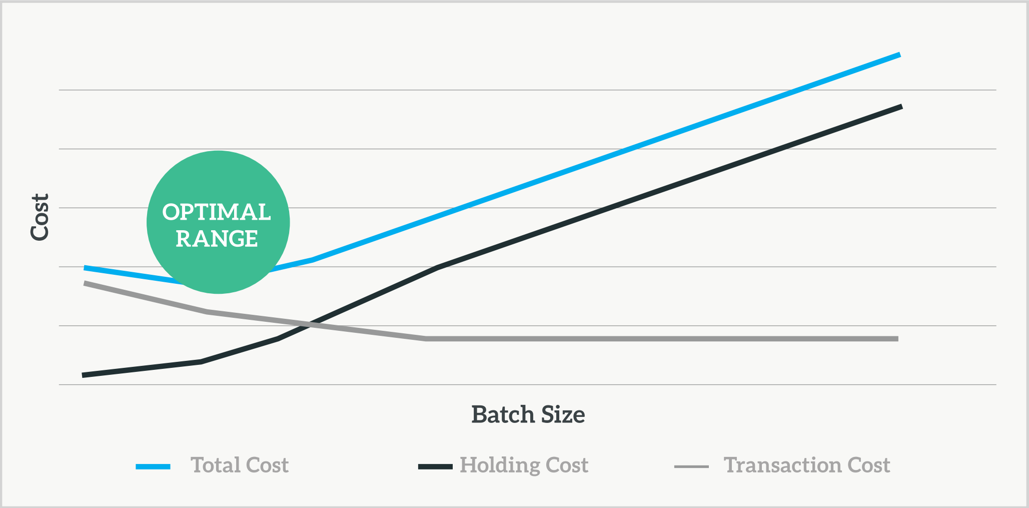
Technology

These investments will improve the optimal Economic Batch Size:

- Decrease transaction costs by automating the fixed costs of a Release: automated build, QA automation, automated deployment, Continuous Integration
- Reduce release Holding Cost by increasing Release frequency, release on every sprint cycle
- Reduce dependencies by decreasing the size (frequency) of the Release (reduce batch size)

The optimal range of the Economic Batch Size in Figure 7 indicates reduced release Transaction Cost, and shorter duration of release cycles (batch size) derived from the investments in automation, quality and decoupling.

Figure 7: Optimized Economic Batch Size



The base camp progression in Figure 8 defines the the investments in automation, quality and decoupling. These investments produce a reduction and elimination of organizational and systemic dependencies. Reducing these dependencies results in the reduction in Time to Value.

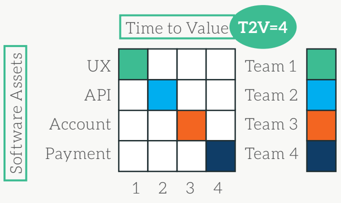
Figure 8: Base Camp Functional Characteristics, Investment and Time to Value

Characteristics | **Investment** | **Time to Value**

Base Camp 1
Predictability
Stable Teams
Backlogs Exist
Delivery Frequently
Measurable Progress
Transparency

- Limited Automation of software engineering practices
- Time to value impeded by team dependencies
- Teams with cross functional skills are aligned to software assets
- Dependencies managed by sequencing work iterations to iteration

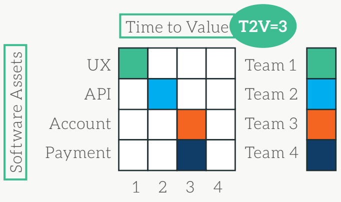
☑ Automated Build
◉ Automated QA Unit Test
□ Automated Environment Build
□ Automated Integration test
□ Automated Deployment
□ Continuous Integration
□ Decouple Monolithic Services
□ Automated Regression Test
□ Microservices
□ Continuous delivery



Base Camp 2
Improve Flow
Quarterly Agile planning
Small Batches
Improved Coordination
Introduce flow based metrics

- Time to value improved by automation of software engineering practices
- Improved prioritization and sequencing of work enables teams to build and test concurrently

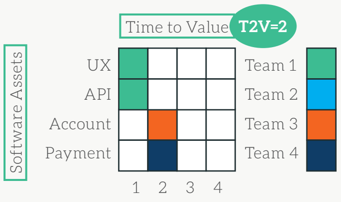
☑ Automated Build
◉ Automated QA Unit Test
◉ Automated Environment Build
◉ Automated Integration test
◉ Automated Deployment
◉ Continuous Integration
□ Decouple Monolithic Services
□ Automated Regression Test
□ Microservices
□ Continuous delivery



Base camp 3
Decoupling
Continuous Integration
Legacy Refactoring
Release Cadence
DevOps

- Improved automation enables teams to begin to align by value
- Decoupling of software platforms reduces dependencies

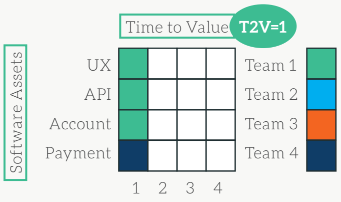
☑ Automated Build
☑ Automated QA Unit Test
☑ Automated Environment Build
☑ Automated Integration test
☑ Automated Deployment
☑ Continuous Integration
◉ Decouple Monolithic Services
◉ Automated Regression Test
◉ Microservices
□ Continuous delivery



Base Camp 4
Autonomous
Adaptive Governance
Economic Prioritization
Continuous Delivery Capable
Release Cadence

- Most or all teams are full cross functional by skills and by value stream and focused on time to value
- Software encapsulation makes services autonomous
- Continuous Delivery capable, but still coupled to Release cadence

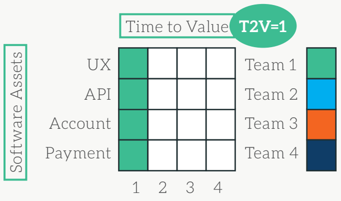
☑ Automated Build
☑ Automated QA Unit Test
☑ Automated Environment Build
☑ Automated Integration test
☑ Automated Deployment
☑ Continuous Integration
☑ Decouple Monolithic Services
☑ Automated Regression Test
◉ Microservices
◉ Continuous delivery



Base Camp 5
Invest to Learn
Outcome based Accountability
Innovation Focused
Design Thinking and Lean Startup
Continuous Delivery

- Software was engineered to be Base Camp 5 ready
- Teams are fully cross functional by skills and by value stream and focused on time to value
- No or minimal dependencies
- Fully automated Cpntinuous Delivery

☑ Automated Build
☑ Automated QA Unit Test
☑ Automated Environment Build
☑ Automated Integration test
☑ Automated Deployment
☑ Continuous Integration
☑ Decouple Monolithic Services
☑ Automated Regression Test
◉ Microservices
☑ Continuous delivery

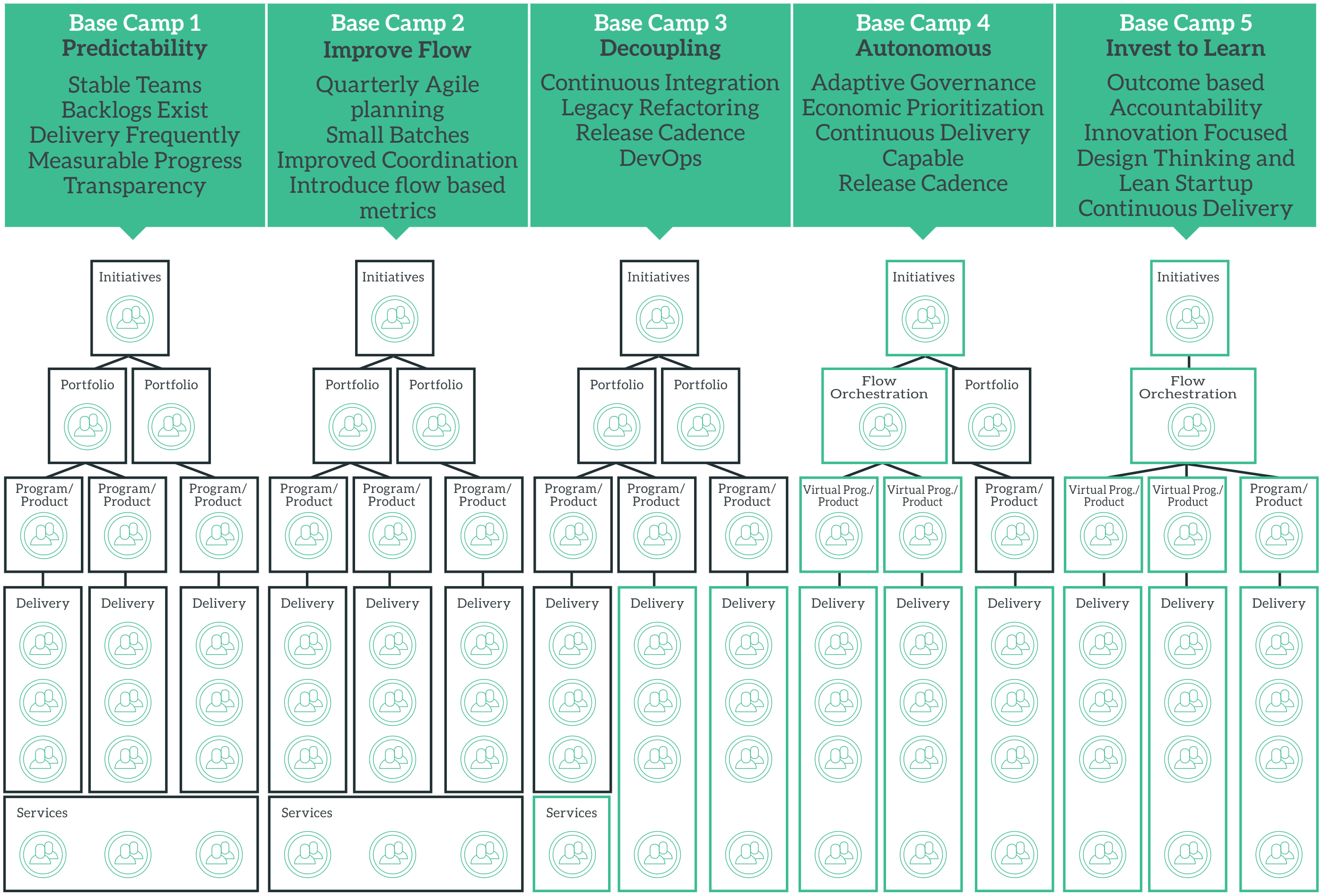


ORGANIZATION EVOLUTION

Figure 9 denotes the organizational changes that occur during the Base Camp progression.

- Base Camp 3 – Services absorbed by Delivery Teams, full functional Value Stream teams
- Base Camp 4-5 – Flow Orchestration, Virtual Program/Product teams, full functional Value Stream teams

Figure 9: Base Camp Organizational Characteristics



Base Camp 1
Predictability
Stable Teams
Backlogs Exist
Delivery Frequently
Measurable Progress
Transparency

Base Camp 2
Improve Flow
Quarterly Agile planning
Small Batches
Improved Coordination
Introduce flow based metrics

Base Camp 3
Decoupling
Continuous Integration
Legacy Refactoring
Release Cadence
DevOps

Base Camp 4
Autonomous
Adaptive Governance
Economic Prioritization
Continuous Delivery Capable
Release Cadence

Base Camp 5
Invest to Learn
Outcome based Accountability
Innovation Focused
Design Thinking and Lean Startup
Continuous Delivery

The Base Camp 1 and Base Camp 2 organization is characterized by:

- Orchestrating organizational and systemic dependencies
- Portfolio Teams to define and prioritize Epics
- Program and Product Teams to define and manage Features and dependencies
- Delivery teams are cross functional by skills and aligned with software assets
- Services teams for shared components and services

The Base Camp 3 organization is characterized by:

- Orchestrating, encapsulating and eliminating organizational and systemic dependencies
- All or most Delivery Teams are fully cross functional by skills and by value stream

In Base Camp 3 the Delivery teams begin assuming responsibility for the development of Services. Code ownership dependencies have been mitigated by Continuous Integration, QA automation, trunk based development, code release readiness standards and peer review by designated codebase custodians.

The Base Camp 4 and Base Camp 5 organization is characterized by:

- Encapsulating and eliminating organizational and systemic dependencies
- Initiative and Epic prioritization
- Introduce flow orchestration to sequence work
- Introduce Virtual Program/Product Teams to elaborate Epics and Features
- Delivery Teams are fully cross functional by skills and by value stream

In Base Camp 4 and 5 flow orchestration is introduced. Organizational and systemic dependencies have been reduced significantly so managing the Portfolio and the sequencing of work is accomplished with a Continuous Flow, Lean Pull model. Minimal team intervention is required. Virtual Program/Product teams are designated for epics and convene for progressive elaboration sessions of the epics and features. Because the dependencies have been significantly reduced it is no longer necessary to sustain a dedicated Program/Product team to manage those dependencies.

BASE CAMP CHARACTERISTICS

Figures 10 and 11 provide a description of Base Camp progressions by focusing on changes in the characteristics for Organization (Figure 9) and Managing the Software Asset (Figure 10).

Figure 10: Base Camp Characteristics – Organization (Predictive-Adaptive)

Characteristics	Base Camp 1	Base Camp 2	Base Camp 3	Base Camp 4	Base Camp 5
Theme	Predictive	Improve Flow	Decoupling	Autonomy	Invest to Learn
Business Value	Predictable Delivery Quality	Quality Improve Flow Throughput	Time to Value	Time to Value	Time to Value
Team Performance	Stable Teams Backlogs Exist Delivery Frequently Measurable Progress Transparency	Quarterly Agile planning Small Batches Manage Flow Flow based metrics	Continuous Integration Legacy Refactoring Release Cadence DevOps	Adaptive Governance Funding Capabilities Financial Controls Economic Prioritization Continuous Delivery capable w/Release Cadence	Outcome based accountability Innovation Focused Design Thinking Lean Startup Continuous Delivery
Organizational	Organize teams to Minimize Organization and Software Dependencies	Orchestrate and Manage Dependencies	Encapsulate and Remove Dependencies	Organize to Optimize Business Value	Execute to Optimize Business Value
Metrics	Backlog Stable Velocity	Lead Time, Cycle Time, Time in Queue, Technical State	Cost of Delay Technical State DevOps Maturity	Time to Business Value DevOps Maturity	Time to Business Value
Measure Outcomes	Based on Post-Release Customer Impact	Based on Post-Release Customer Impact	OKR’s, KPI’sInstrument to measure outcomes Real Time, during and post Release Implementation	OKR’s, KPI’s Instrument to measure outcomes Real Time, during and post Release Implementation	OKR’s, KPI’s Instrument to measure outcomes Real Time, Continuous

Figure 11: Base Camp Characteristics – Managing the Software Asset (Emergent-Convergent)

Characteristics	Base Camp 1	Base Camp 2	Base Camp 3	Base Camp 4	Base Camp 5
Theme	Predictive	Improve Flow	Decoupling	Autonomy	Invest to Learn
Software Assets	Technical Debt	Feature Toggles Technical Debt	Refactoring Reduce or Eliminate Dependencies DevOps Compatible Dependencies DevOps Compatible	DevOps Autonomous Services Micro-services	Adaptive Services
Feature Toggles	No	Inconsistent	Managed	Managed	Managed
Code Branch Strategy	Feature and Release	Feature and Release	Release and Trunk	Trunk	Trunk
Software Engineering Standards such as TDD, SOLID, DRY (within services)	No	Inconsistent	Yes	Yes	Yes
Automated Build & Merge Frequency	2-3 times per Week	Daily	Continuous	Continuous	Continuous
Continuous Integration	No	Daily	Yes, Every Build	Yes, Every Build	Yes, Every Build
Unit Testing Automation	Inconsistent	Automated for all new software	Automated for all new and updated software	Sufficient Automation for Continuous Delivery	More than sufficient automation for Continuous Delivery
Validation Test Automation	Inconsistent	Automated for all new software	Automated for all new and updated software Code Coverage based on Complexity and Volatility	Sufficient Automation for Continuous Delivery	More than sufficient automation for Continuous Delivery
Static Code Analysis	Periodic review	Periodic review	No increase in complexity For each iteration	No increase in complexity For each build	No increase in complexity For each build
Automated Deployment	No	Inconsistent	Yes	Yes	Yes
Notification and Reporting of Build Results	No	Inconsistent	Yes	Yes	Yes
Defect Prevention, Build Accelerate	No	No	Inconsistent	Yes	Yes
Change Implementation	Partially Automated, Release Schedule	Partially Automated, Release Schedule	Automated, Release Schedule	Automated, A/B Testing, Blue-Green Release, Canary, Release Schedule	Automated, A/B Testing, Blue-Green Release, Canary, Continuous
Continuous Delivery	No	No	No	Capable, but dependencies require Release Schedule dependencies require Release Schedule	Yes

CONCLUSION

- All business organizations have distinct organizational and systemic dependencies derived from their evolution over time.
- The cause and effect of organizational structure and system design are inter-twined as indicated by Conway's law.
- It is necessary to assess the current state, define the end state goals, and then to provide the roadmap that will facilitate the progression from current to goal state.
- The Agile adoption governance model is an enabler of organizational and systemic transformation.
- Agile adoption frameworks are designed and optimized to solve a specific problem.
- As the organization's Agile adoption progresses, the problems to solve change also.
- The Agile adoption governance model must be adaptive to the planned reductions in organizational and systemic dependencies.