build better systems faster

INDUSTRIAL DEVOPS

Dr. Suzette Johnson
and **Robin Yeman**

Forewords by **Mik Kersten** and **Dean Leffingwell**

IT Revolution
Independent Publisher Since 2013
Portland, Oregon

IT Revolution
25 NW 23rd Pl, Suite 6314
Portland, OR 97210

First Edition
Printed in the United States of America
28 27 26 25 24 23 1 2 3 4 5 6 7 8 9 10

Cover design by D.Smith Creative, LLC
Book design by D.Smith Creative, LLC

Library of Congress Control Number: 2023027080

ISBN: 9781950508792
eBook ISBN: 9781950508808
Web PDF ISBN: 9781950508815
Audio: 9781950508822

INDUSTRIAL DEVOPS

CONTENTS

FIGURES & TABLES

Chapter 9

Chapter 10

Chapter 11

Chapter 12

Chapter 13

Appendix C

PROJECT SUCCESS RATES

METHOD	SUCCESSFUL	CHALLENGED	FAILED
Agile	42%	47%	11%
Waterfall	13%	59%	28%

Figure 0.1: Agile vs. Waterfall

Source: Anthony Mersino, "Why Agile Is Better than Waterfall (Based on Standish Group CHAOS Report 2020)."

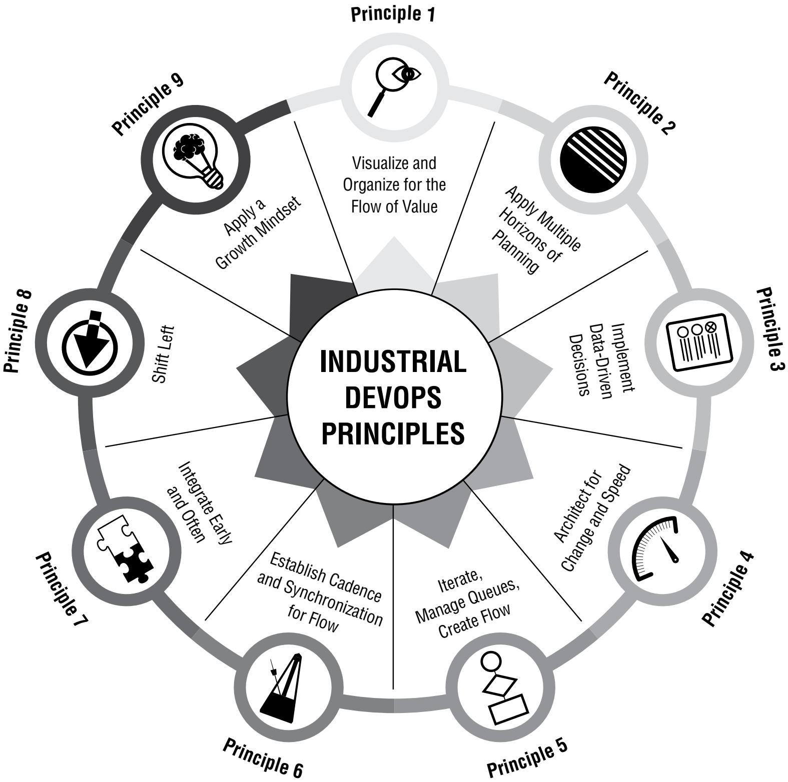


Figure 1.1: Industrial DevOps Principles

Misconceptions about Industrial DevOps
Agile/DevOps development efforts don’t plan.
Agile/DevOps programs constantly change, and that doesn’t work for hardware.
Agile/DevOps does not have systems engineering practices.
Agile/DevOps programs sacrifice quality for speed.
Agile/DevOps does not have any documentation.
Agile/DevOps is only for teams, not managers/leaders.
Agile/DevOps requires deploying operations continuously.
Agile/DevOps requires everyone to be colocated.
Agile/DevOps practices are only for software.
Agile/DevOps does not work with safety-critical systems.
Agile/DevOps requires you to complete a whole system in two weeks.

Table 3.1: Misconceptions about Industrial DevOps

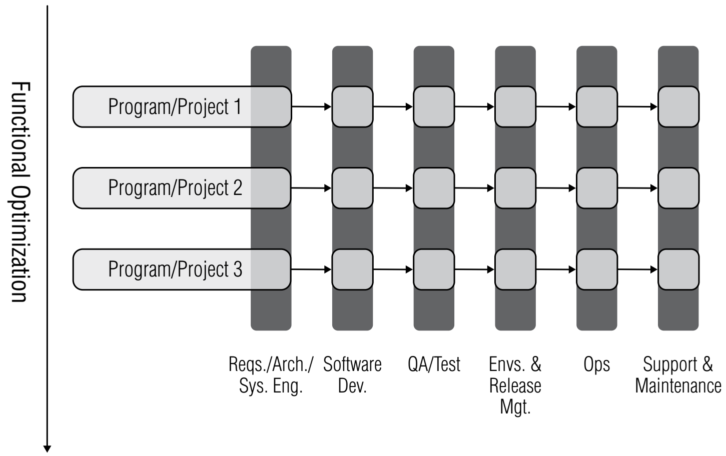


Figure 4.1: Functional Organizational Structure

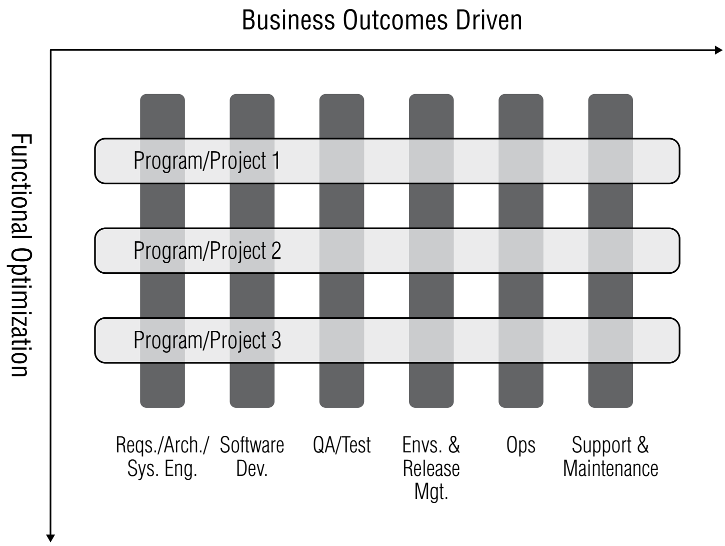


Figure 4.2: Matrixed Organizational Structure

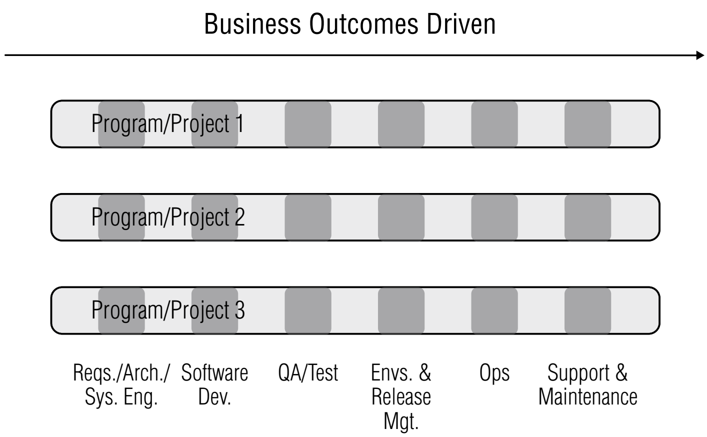


Figure 4.3: Divisional Organizational Structure

Table 4.1: Four Team Types of a Satellite System

Team Topologies Team Type	Description	Satellite System Team
Stream-Aligned Team	Aligned to a flow of work from (usually) a segment of the business domain.	Payload team
Enabling Team	Helps a stream-aligned team to overcome obstacles. Also detects missing capabilities.	Cyber security team
Complicated-Subsystem Team	Where significant mathematics/ calculation/technical expertise is needed.	Guidance, navigation, and control Team
Platform Team	A grouping of other team types that provide a compelling internal product to accelerate delivery by stream-aligned teams.	Continuous delivery pipeline team

Table 4.2: Three Interaction Modes of a Satellite System

Mode of Interaction	Description	Satellite System Example
Collaboration	Working together for a defined period of time to discover new things (APIs, practices, technologies, etc.).	The payload team collaborates with the guidance, navigation, and control Team to transmit navigation signals over S Band.
X-as-a-Service	One team provides and one team consumes something "as a service."	The guidance, navigation, and control team can provide navigation data as a service to other components.
Facilitation	One team helps and mentors another team.	The thermal team mentors the structures team.

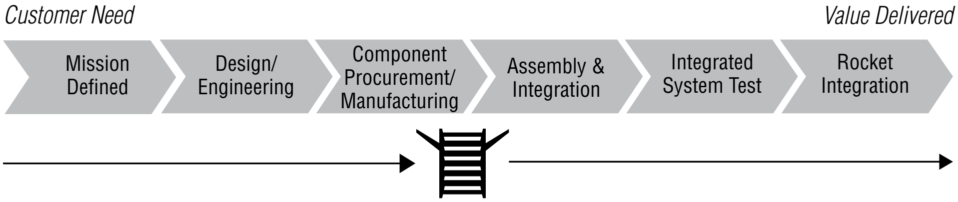


Figure 4.4: CubeSat Value Stream

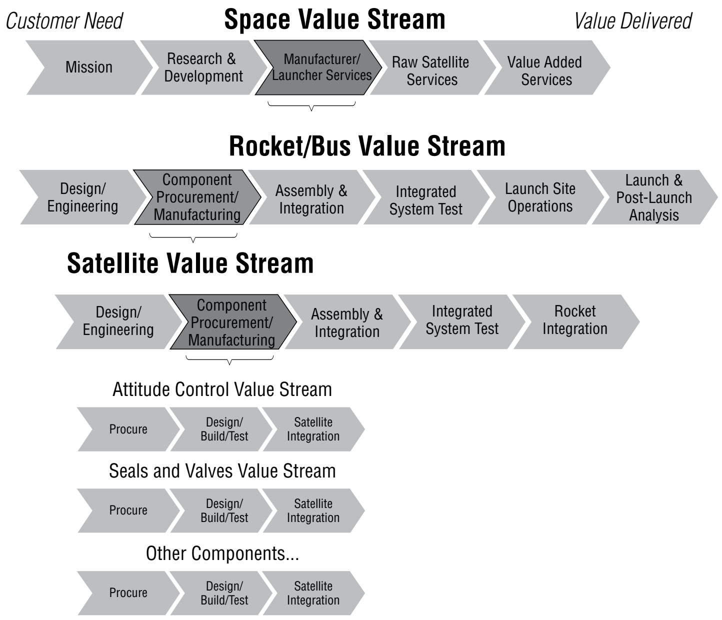


Figure 4.5: Example of Nested Value Streams for CubeSat Constellation

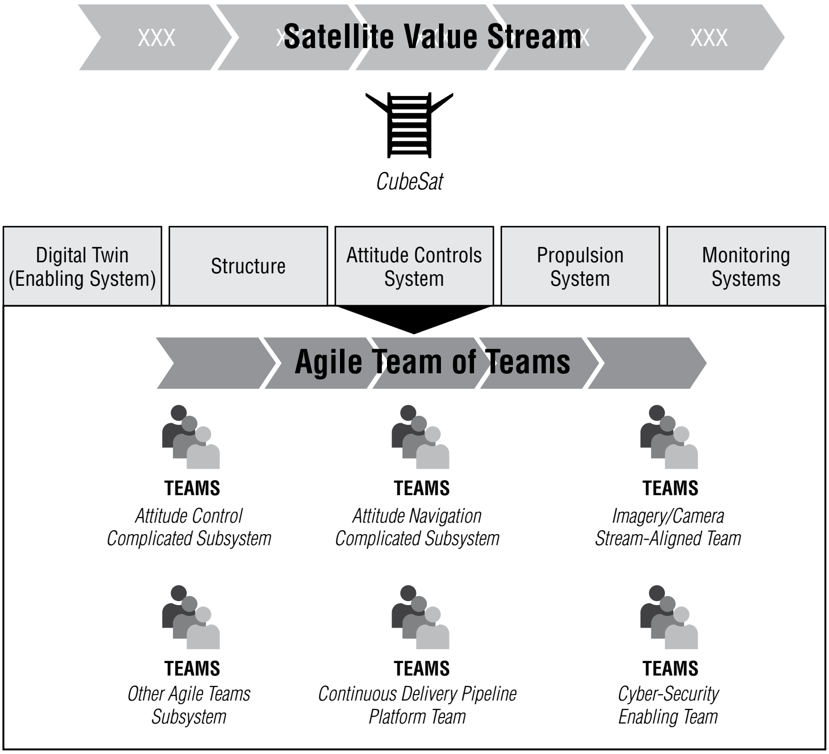


Figure 4.6: Attitude Control System Team-of-Teams Structure

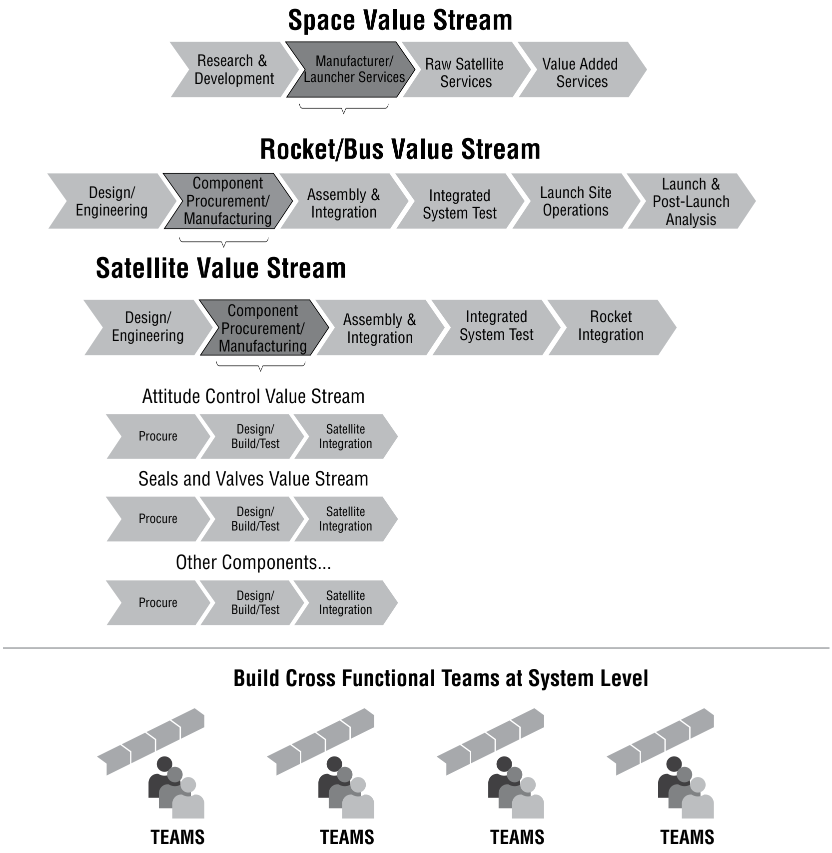


Figure 4.7: Nested Value Stream with Cross-Functional Teams

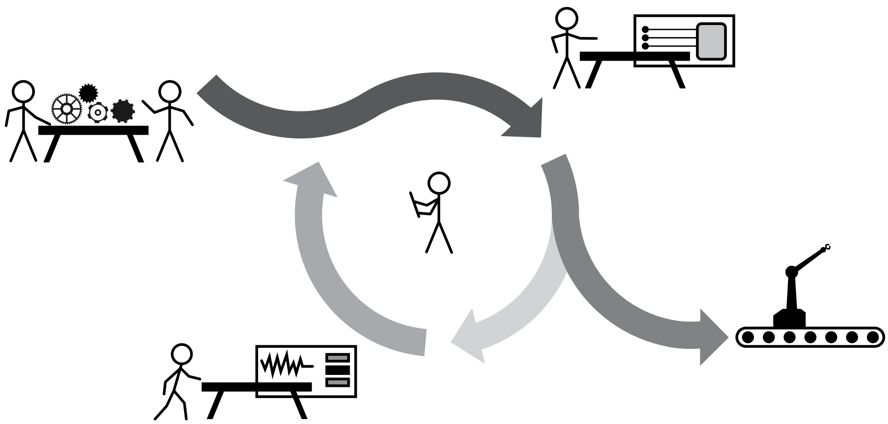


Figure 4.8: Manufacturing Floor

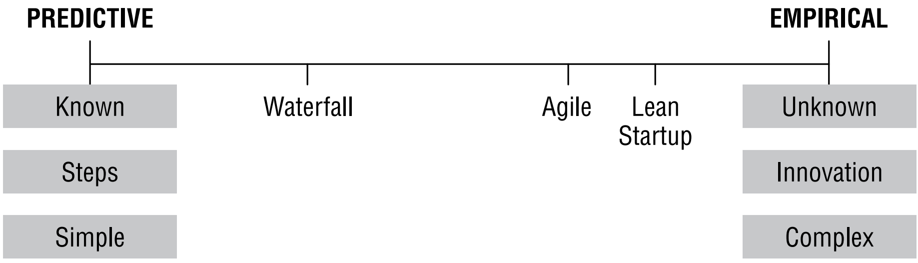


Figure 5.1: Predictive vs. Empirical Process Control

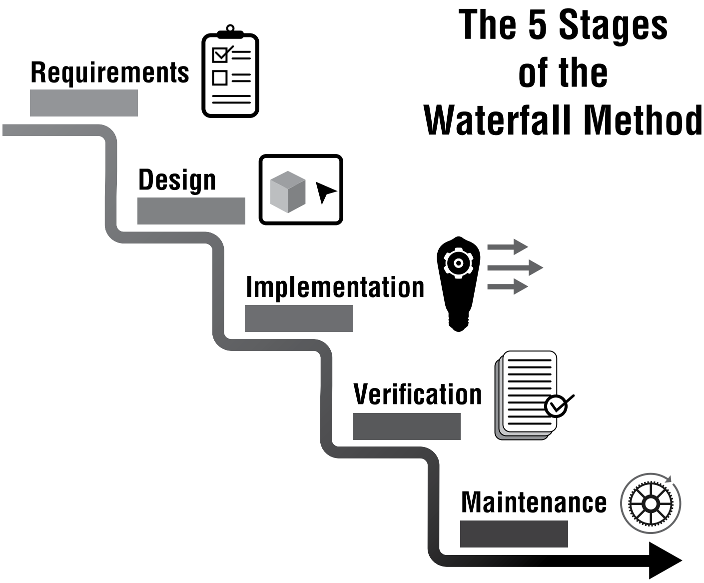


Figure 5.2: Royce and the Steps for Solutioning a System

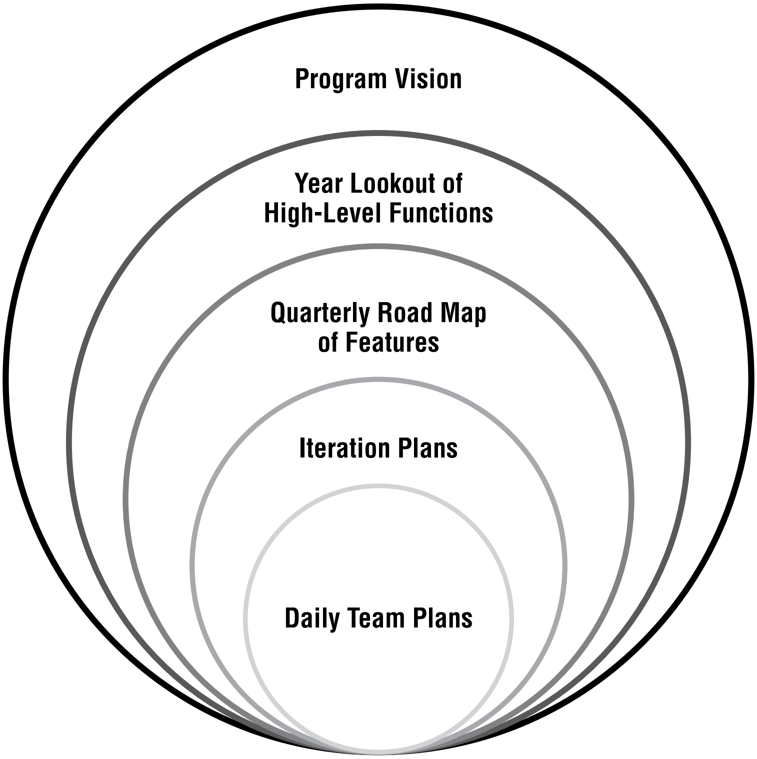


Figure 5.3: Multiple Horizons of Planning

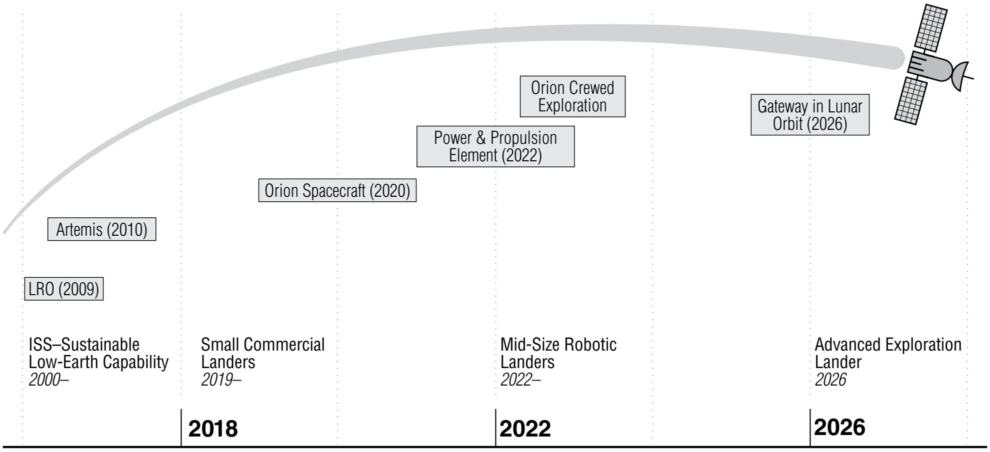


Figure 5.4: NASA's Road Map for Human Space Exploration

Source: Adapted from Foust, "NASA Roadmap Report Provides Few New Details on Human Exploration Plans," SpaceNews. September 25, 2018. https://spacenews.com/nasa-roadmap-report-provides-few-new-details-on-human-exploration-plans/

	Time	Scope
1.	Program/product plan (entire time box)	Product vision
2.	Multiyear plan (1–5-year time box)	Epic (business outcome)
3.	Annual plan (1-year time box)	Epic (business outcome)
4.	Quarterly plan (12–13 weeks)	Feature (business outcome that fits within quarter)
5.	Iteration plan (1–4 weeks)	User story (user outcome that fits in iteration)
6.	Daily plan (8 hours)	Task (individual outcome for today)

Table 5.1: Time and Scope Defined

Table 5.2: Patterns for Decomposition

Pattern		Scope
1.	Work flow steps	Break out all of the steps of the work flow required to deliver value
2.	Business rule variations	Accomplishment of different business rules
3.	Major effort	Large-effort items can often be split, where the first one is the instantiation of capability and the remaining continue to improve
4.	Simple/complex	Capture simplest version of feature and complete remaining to add complexity
5.	Variations in data	Data variations, such as data sources, complexity, language variants
6.	Data methods	Split by the user interface itself
7.	Deferring system qualities	Begin with a simple capability and add the system qualities incrementally
8.	Operations	Order of operations, such as CRUD (create, read, update, delete)
9.	Use case scenarios	Split by goals or scenarios

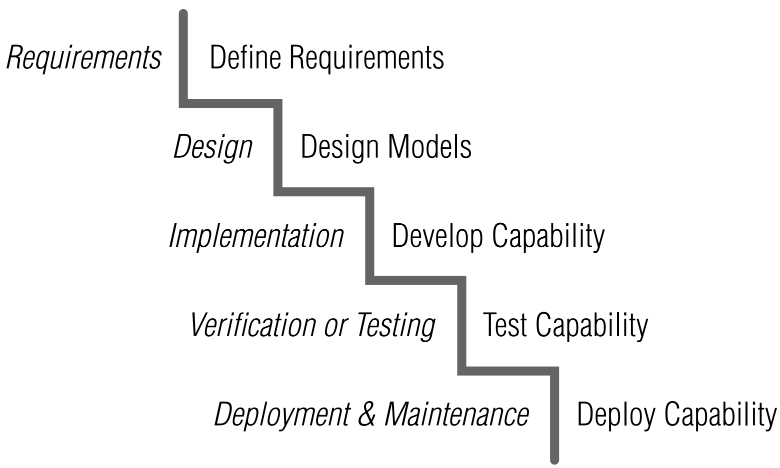


Figure 5.5: Big Bang Solution

Business Problem/Mission Need (1)

What business have you indentified that needs help?

There is a growing demand for improved weather data with reduced costs and increased production rate.

Solution Ideas (5)

List product, feature, or enhancement ideas that help your target audience achieve the benefits they're seeking

Start in a smaller niche area—impove weather forecasting accuracy and data analystics. Iteratively launch 200 CubeSats over 15 months and improve operations to reduce lead time to 12 months or lower; Low Earth Orbit solution.

Business Outcomes (2)

What changes in customer behavior will indicate you have solved a real problem in a way that adds value to your customers?

Starting with our CubeSat weather mission we will build from this experience as we scale with reduced costs and faster time to market.

Users & Outcomes (3)

What types of users and customers should you focus on first?

Weather forecast customers who want better, more accurate data to improve decision-making and reporting.

User Benefits (4)

What are the goals our users are trying to achieve? What is motivating them to seek out your solution? (e.g., do better at my job OR get a promotion)

Provide timely weather imagery data and broaden the coverage so that they can provide more accurate reports to their user community.

Hypotheses (6)

Combine the assumptions from 2, 3, 4 & 5 into the following template hypothesis statement:

"We believe that [business outcome] will be achieved if [user] attains [benefit] with [feature]."

Each hypothesis should focus on one feature.

We believe that we will be able to scale our business if our weather forecast customers provide more accurate reports to their user community by receiving more timely data with wider coverage.

What's the most important thing we need to learn first? (7)

For each hypothesis, identify the riskiest assumption. This is the assumption that will cause the entire idea to fail if it is wrong.

Performance constraints of the current design; limitations of imagery/ resolution.

What's the least amount of work we need to do to learn the next most important thing? (8)

Brainstorm the types of experiments you can run to learn whether your riskiest assumption is true or false.

Use of models for early-stage design study and simulation of the CubeSat mission.

Figure 5.6: Lean UX Canvas Example: CubeSat

Source: Lean UX Canvas template is used with permission. Jeff Gothelf and Josh Seiden, *Lean UX: Designing Great Products with Agile Teams*, 3rd Edition (O'Reilly Media, 2021).

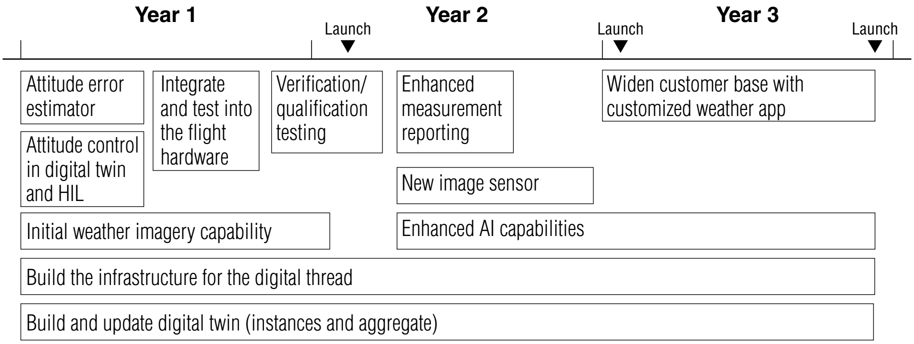


Figure 5.7: Sample Road Map for CubeSat Mission

Quarter 1	Quarter 2	Quarter 3	Quarter 4
Feature: Develop attitude error estimator and integrate into the processor-in-the-loop environment **Feature:** Build the infrastructure for digital thread	**Feature:** Incorporate attitude control algorithms in partial digital twin & hardware-in-the-loop	**Feature:** Develop image-processing improvements and test using the camera PIL environment	**Feature:** Integrate and test image-processing updates into the flight hardware

Figure 5.8: Annual Road Map Broken into Quarters for CubeSat

Table 5.3 Feature Example with Acceptance Criteria

Feature	Acceptance Criteria
Determine attitude error estimator and integrate into processor-in-the-loop environment.	Incorporate into the target software environment; demonstrate on a processor-in-the-loop environment/simulation (e.g., hybrid cyber-physical twin, emulated processor/other subsystems). Demonstrate the feed to the attitude controller.

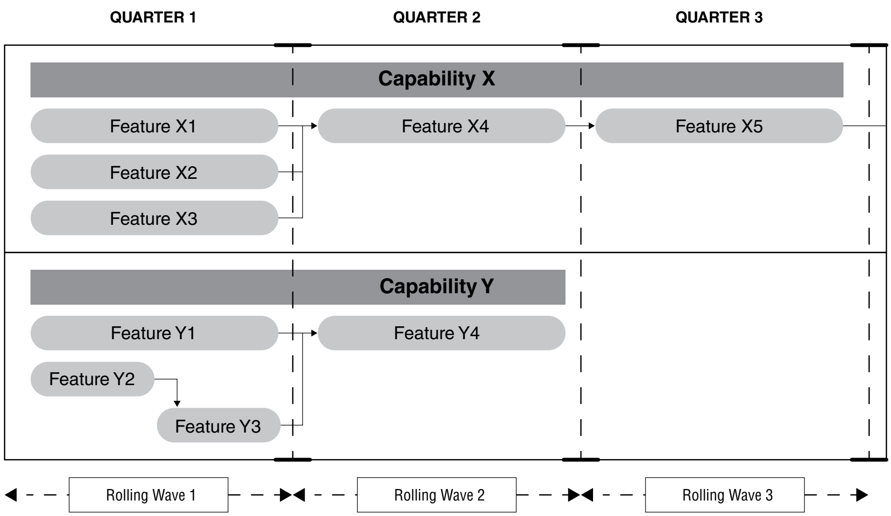


Figure 5.9: Rolling-Wave Planning Example

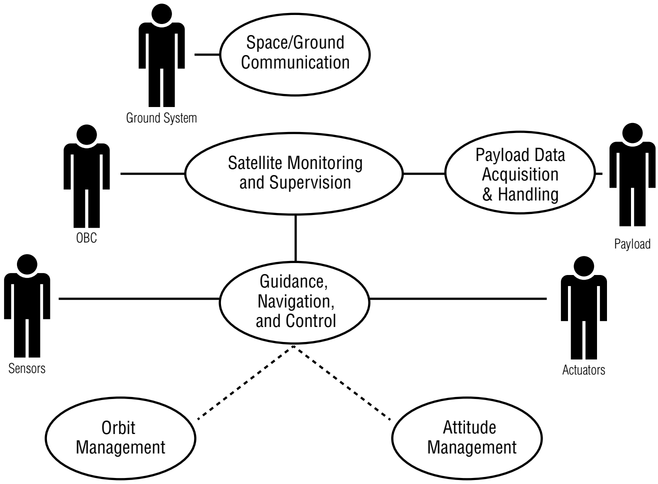


Figure 5.10: CubeSat Space Ground Communication Use Case

Quarter 1 Sample

Feature	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	Iteration 6
Develop attitude error estimator and integrate into the processor-in-the-loop environment.	Verify that attitude sensor measures current state and feeds data to attitude navigation.	Generate state estimates based upon attitude sensor measurements in SIL.	Determine attitude error using state estimate and desired state from navigation in SIL.	Perform characterization of attitude error estimation in the SIL environment.	Port attitude estimation software into the PIL environment.	Perform characterization of attitude error estimation in the PIL environment and validate alignment with the SIL environment.
Build the infrastructure for the digital thread.	Define the architecture of the digital thread (MVP).	Set up of modeling tool and interface with backlog; communicate process.	Build out connection to requirements.	Add change management process.	Create bill of materials (BOM) structure.	Ensure interfaces and traceability.

Figure 5.11: Quarterly Road Map: CubeSat Example

Table 5.4: Example of an Iteration Backlog Item for Attitude Controller

Feature	Iteration 1
Determine attitude error so that it may be fed to the attitude controller	**Estimate:** <relative size> (e.g., 5 story points) **Story (1)**: As an Attitude Sensor, I want to measure the current state of attitude so that I can adjust the attitude. **Acceptance Criteria:** 1. Identify error source inputs for the attitude estimator. 2. For each attitude error source, identify the software elements that will need to be modified to provide error estimates to the controller. 3. Update simulation elements for contributors to adjust attitude error. 4. Model changes in software-in-the-loop (SIL) and hardware-in-the-loop (HIL). 5. Perform Monte Carlo assessment of attitude adjustment error estimator in SIL and HIL.

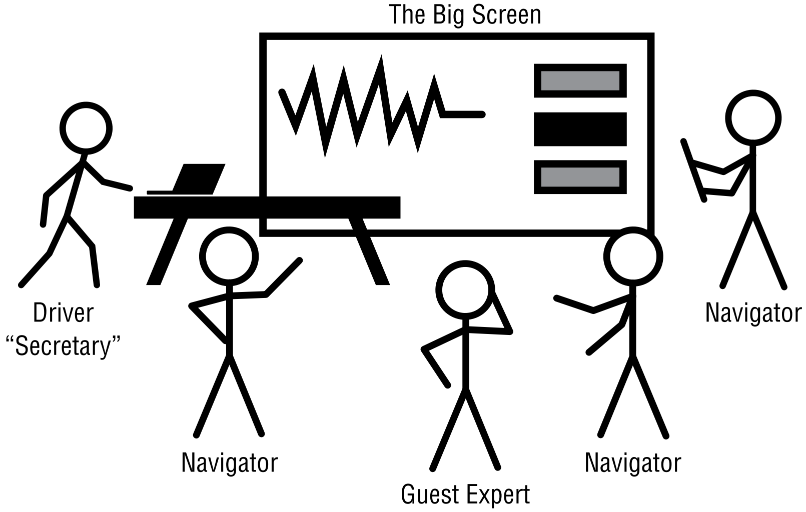
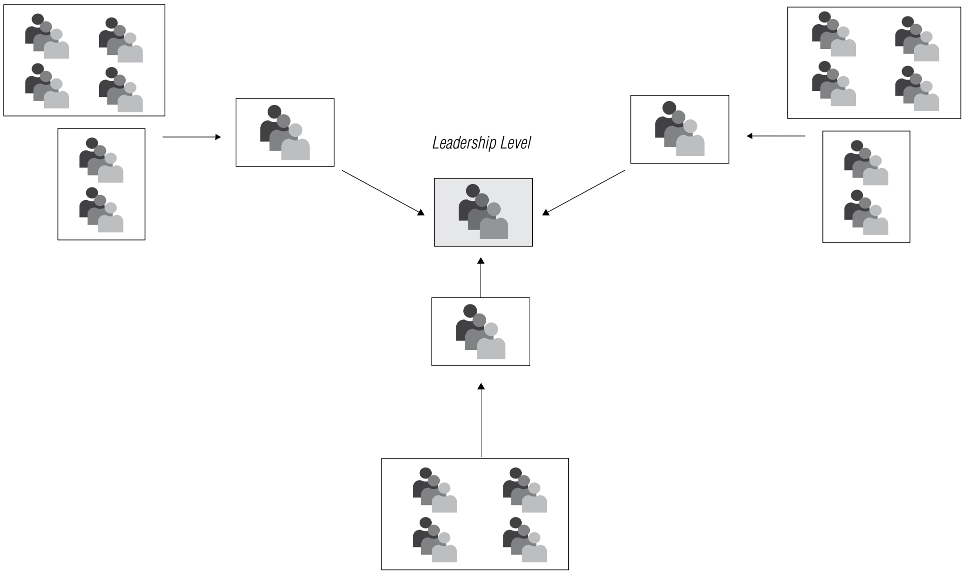


Figure 5.12: Mob Work



Communication path from teams to leadership. Impediments reach management level every day.

Figure 5.13: CubeSat Team of Agile Teams

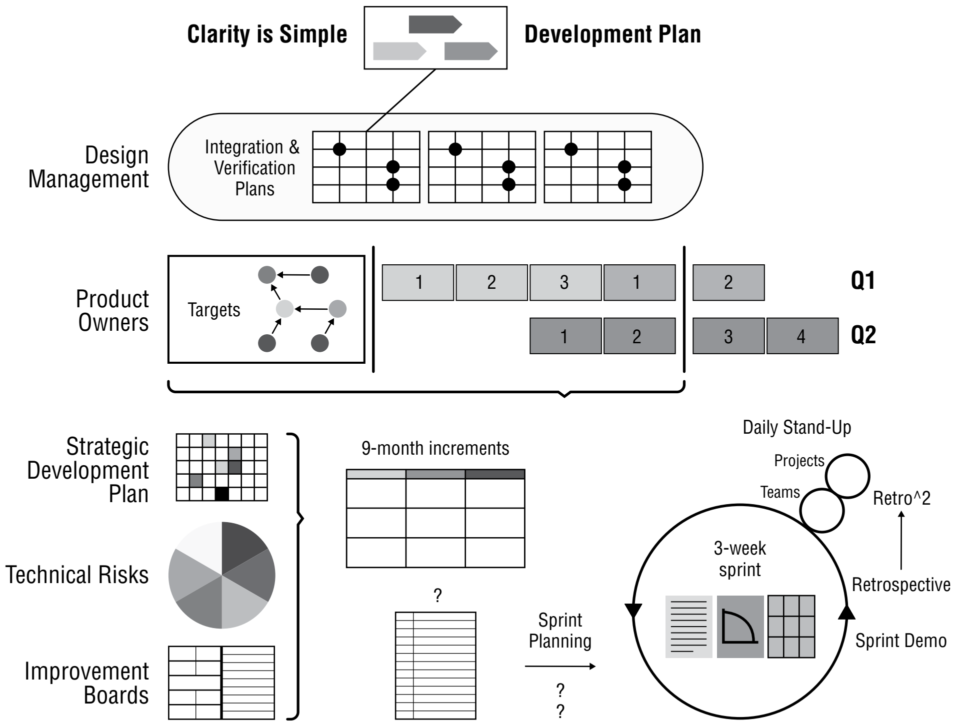


Figure 5.14: Saab Gripen Process

Source: Joe Justice et al., "Owning the Sky/SAAB"

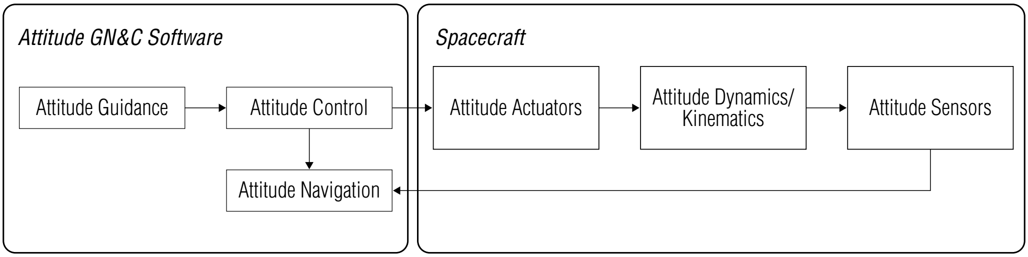


Figure 6.1: Attitude Guidance, Navigation, and Control

Source: Steve Ulrich, professor at Carleton University in Ottawa, Canada, and the founding director of the Spacecraft Robotics and Control Laboratory.

Table 6.1: Example Backlog with Objective Evidence for Demonstration

Backlog Item	Time Horizon	Description of What is Needed	Objective Evidence and How It's Demonstrated
Epic	>1 Quarter	Implement attitude controller	Incorporate into hardware-in-the-loop closed-loop simulation (e.g., partial physical twin).
Feature	Quarter	Determine attitude so that we can adjust the attitude.	Incorporate into the target software environment; demonstrate on a processor-in-the-loop environment/simulation—hybrid—cyber-physical twin, emulated processor/other subsystems. Demonstrate the adjustment made in the attitude controller.
Story	Iteration	Verify that attitude sensor measures current state and feeds data to attitude navigation.	Incorporate into software-in-the-loop simulation for the attitude controller. Demonstrate with the digital twin/model (the digital twin is still being built out).

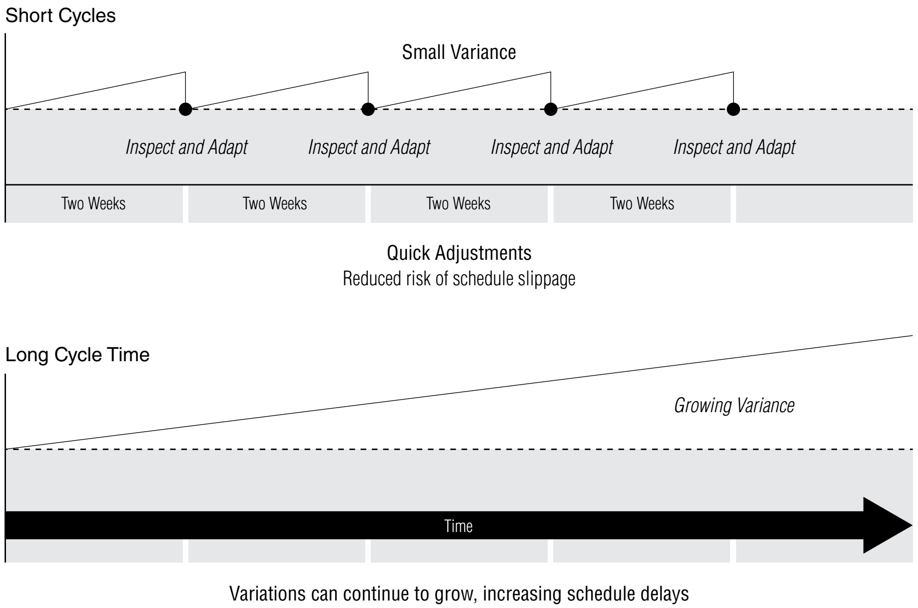


Figure 6.2: Iterative Development Provides Fast Feedback and Decreases Deviations

Leading Indicators

Measures when driving toward a desired business outcome

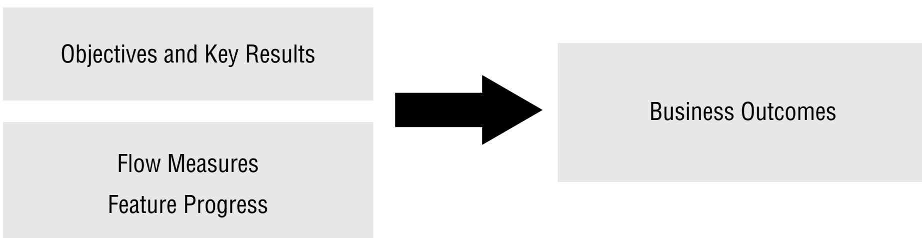


Figure 6.3: Leading Indicators of Business Outcomes

Table 6.2: Measures of the Flow of Value

Measure	Description
Flow time	Measures time to market; namely the time elapsed from "work start" to "work complete" on a given flow item, including both active and wait times."[13]
Flow efficiency	Is the ratio of active time out of the total flow time.
Flow velocity	Is the number of items being completed over a defined unit of time. In our context, flow velocity is measured using story points and is the range of story points a team delivers over several iterations.
Flow load	Measures the number of flow items currently in progress (active or waiting) within a particular value stream.
Flow distribution	Measures the distribution of the four flow items—features, defects, risks, and debts—in a value stream's delivery.
Work in progress (WIP)	WIP can be thought of as the functionality, architecture, or inventory work that is in progress but not yet completed.

Table 6.3: Lead Time vs. Cycle Time

Measure	Description
Lead time (production lead time)	"The time required for a product to move all the way through a process or a value stream from start to finish. At the plant level, this often is termed door-to-door time. The concept also can be applied to the time required for a design to progress from start to finish in product development or for a product to proceed from raw materials all the way to the customer,"[15] as defined by Lean.Org/Lean Enterprise Institute. Lead time is the period between the start of a process and its conclusion. That is, it's the amount of time it takes to make a product or service so it's usable for the customer.[16]
Cycle time	"Time required to produce a part or complete a process, as timed by actual measurement,"[17] as defined by Lean.Org/Lean Enterprise Institute.

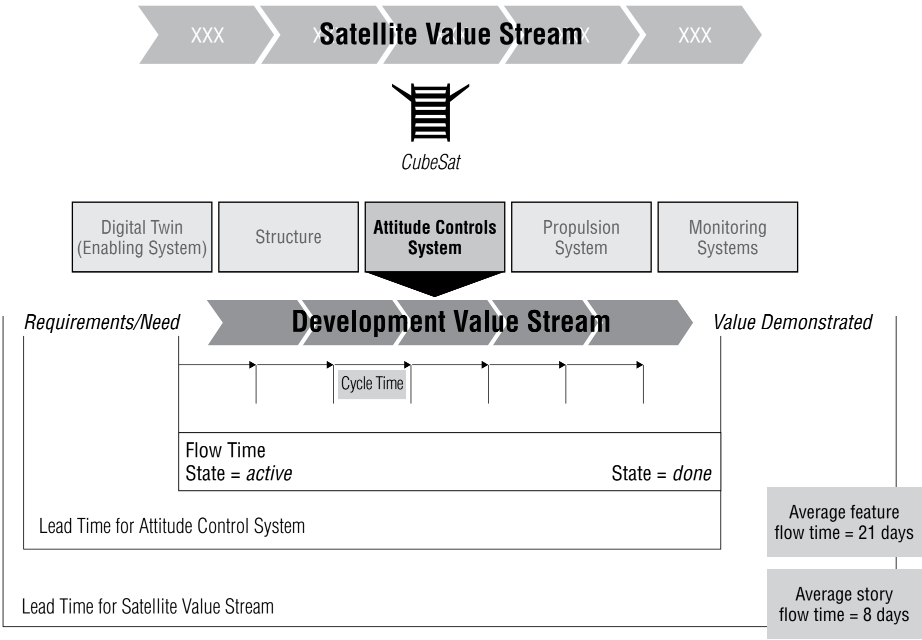


Figure 6.4: Flow Time Illustrated

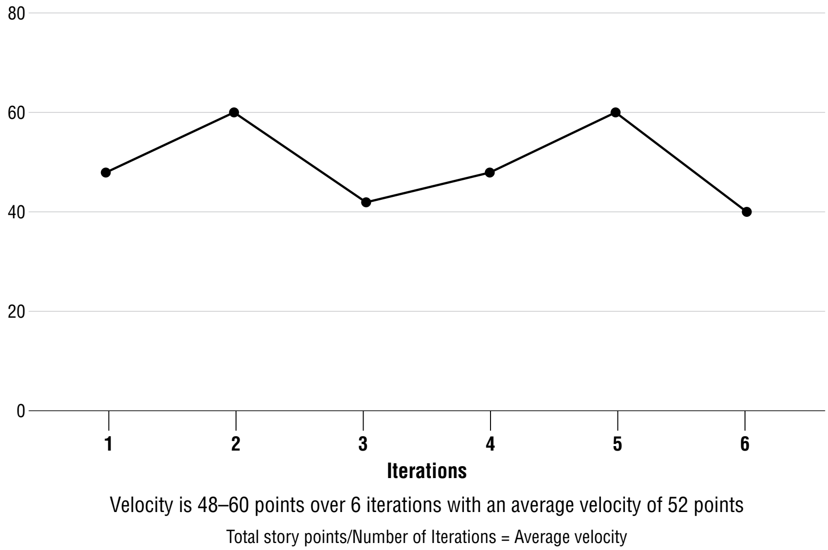


Figure 6.5: Flow Velocity for a Team

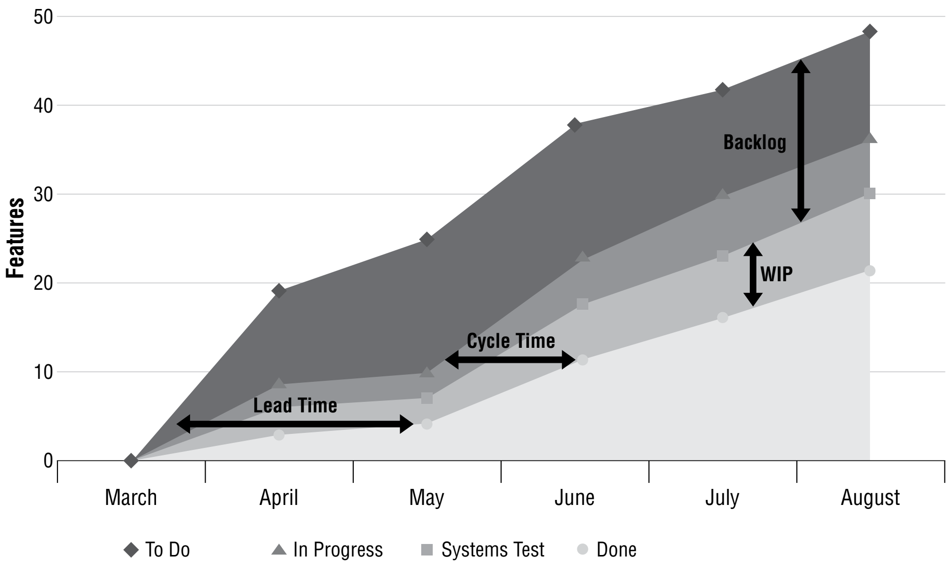


Figure 6.6: Cumulative Flow Diagram

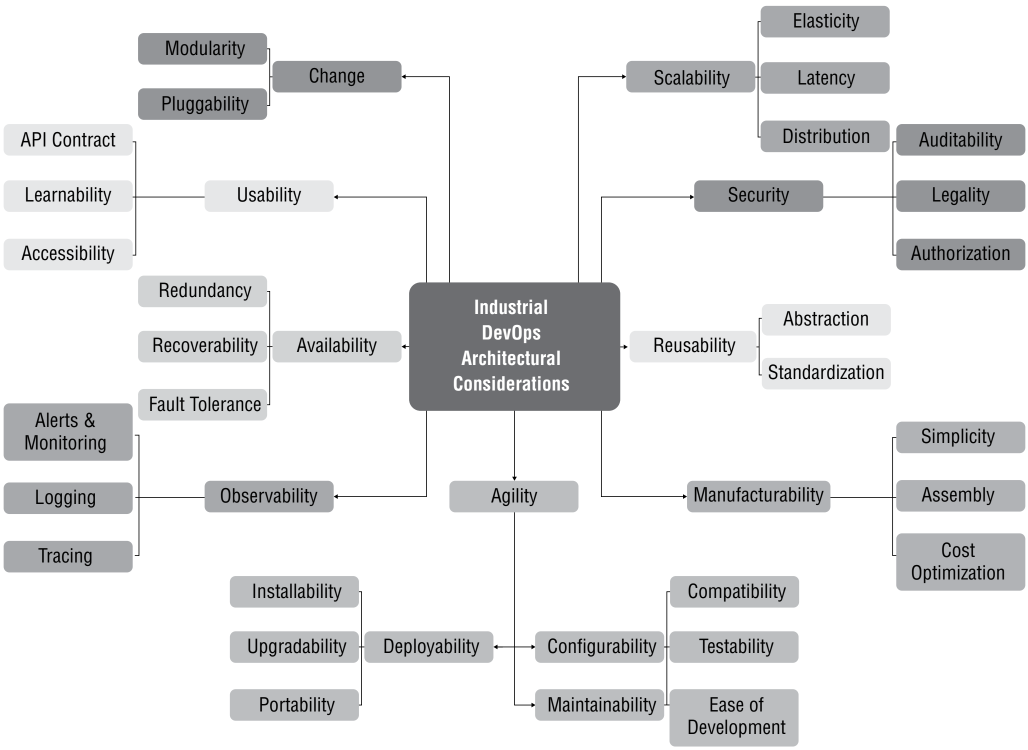


Figure 7.1 Architectural Considerations for Industrial DevOps

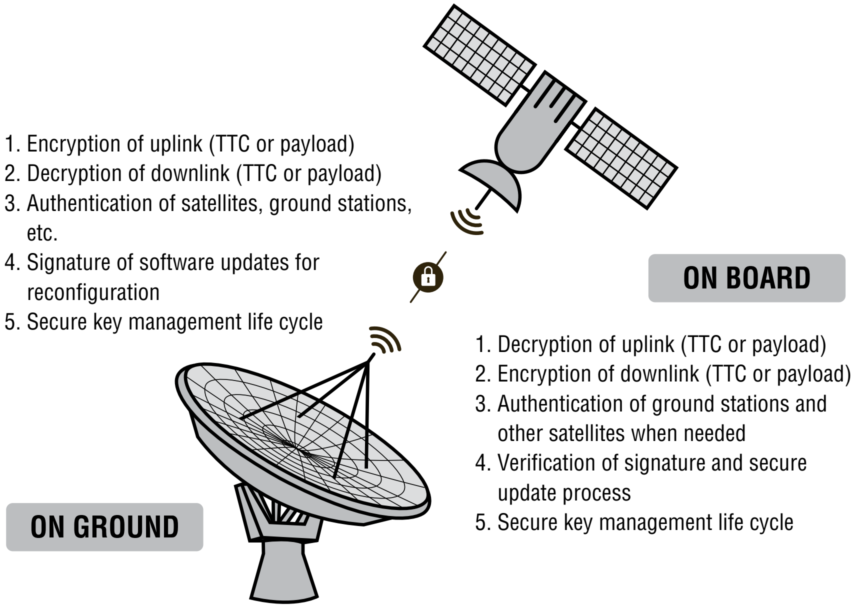


Figure 7.2: Security Architecture for CubeSat

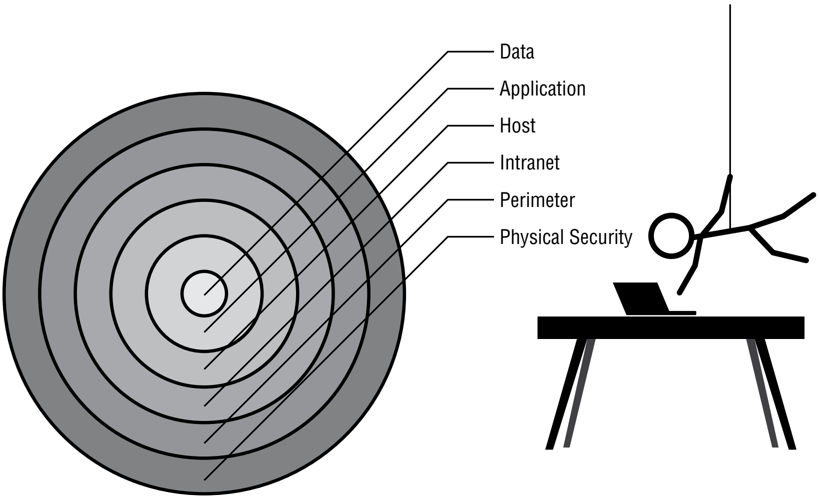


Figure 7.3: Defense in Depth

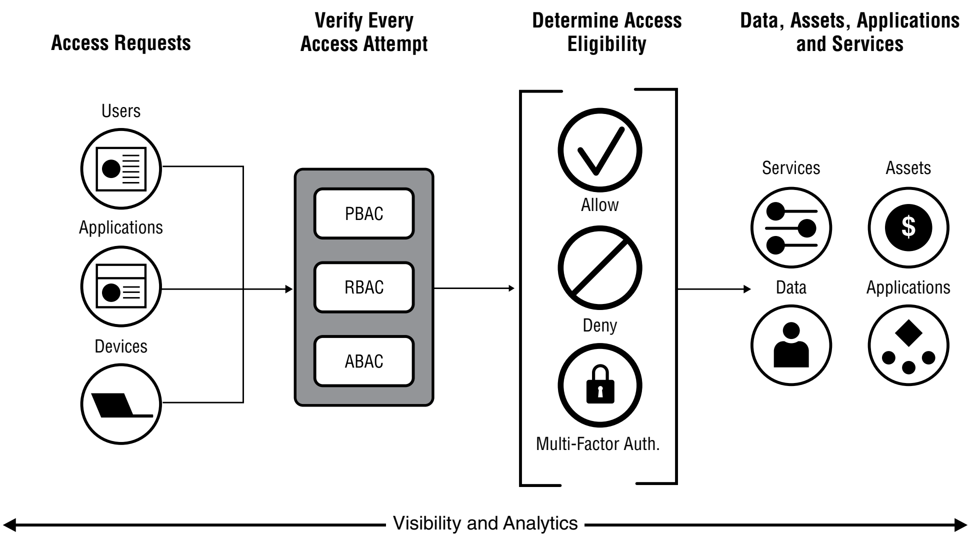


Figure 7.4: Zero-Trust Architecture

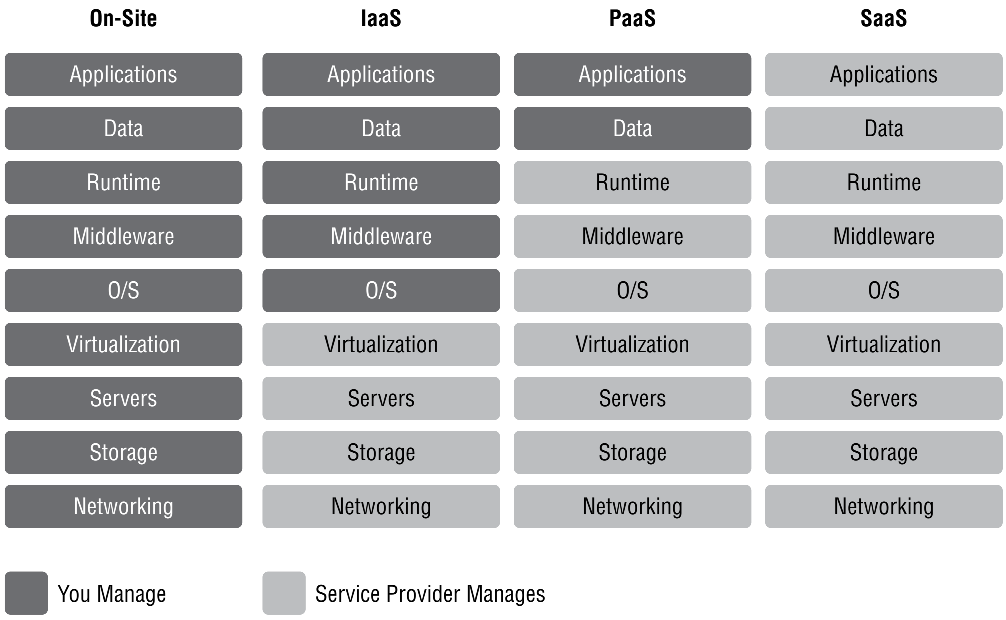


Figure 7.5: The Journey from Fully On-Premises Solutions to Fully Off-Premises Solutions

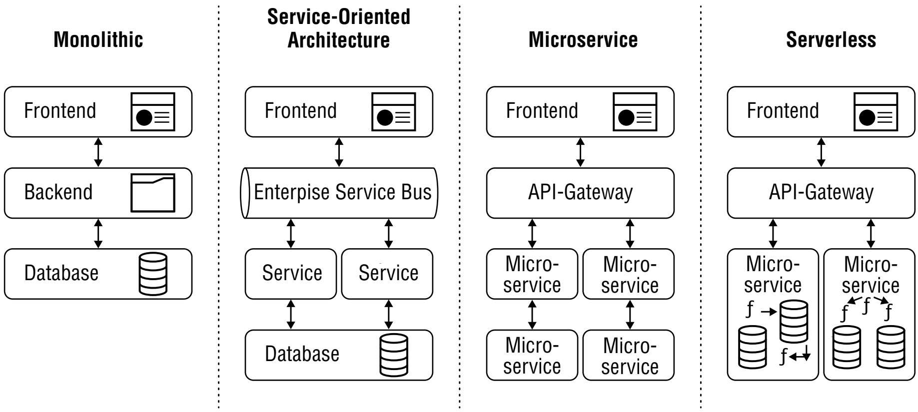


Figure 7.6: The Evolution of Architectures

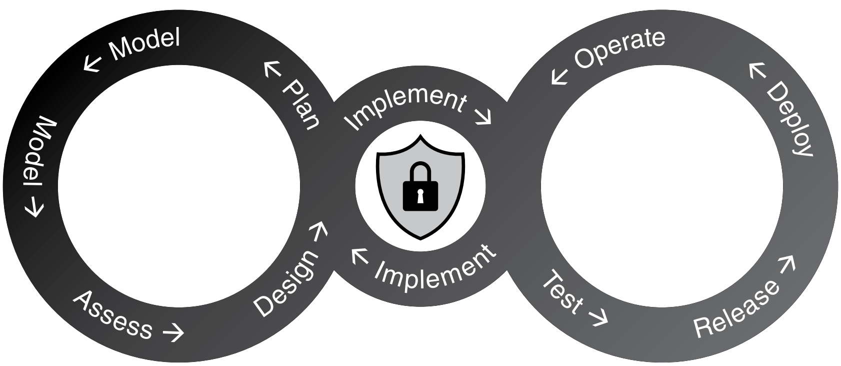


Figure 7.7: TwinOps

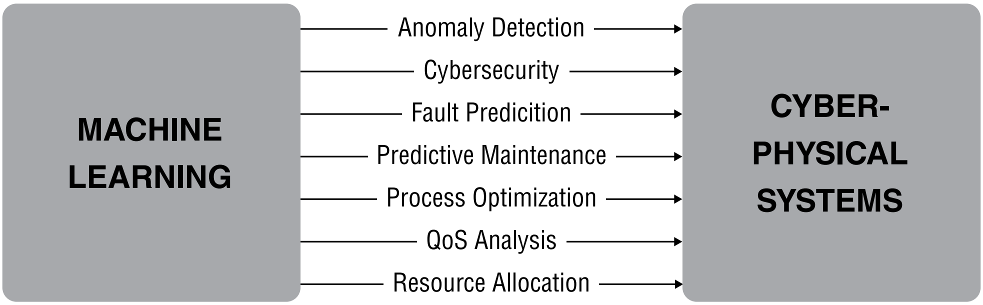


Figure 7.8: AI/ML in Cyber-Physical Systems

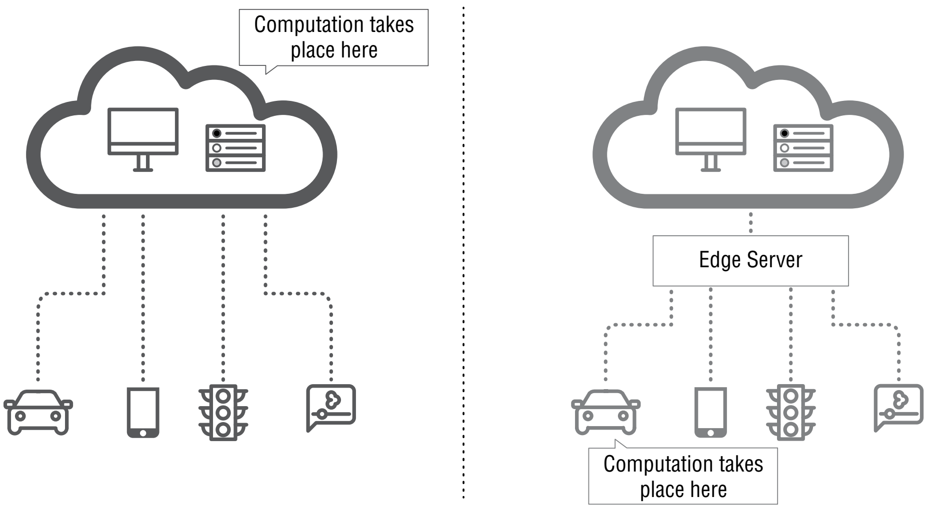


Figure 7.9: Cloud Computing vs. Edge Computing

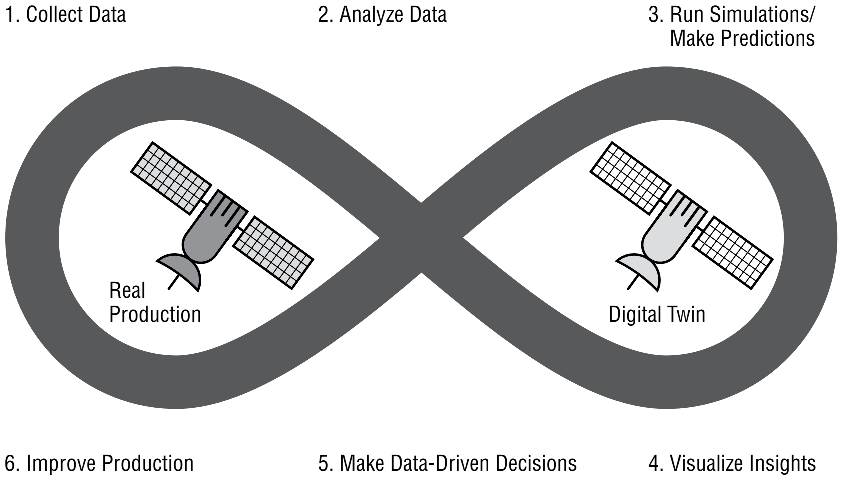


Figure 7.10: Digital Thread

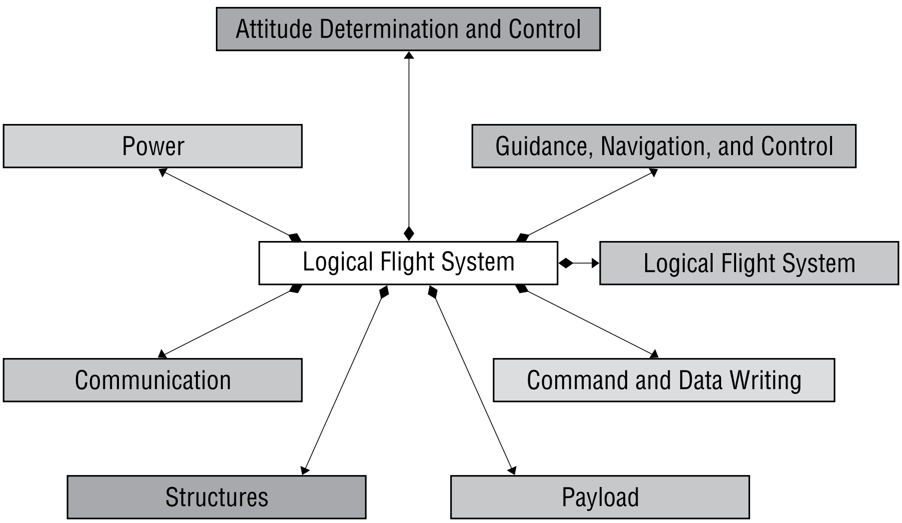


Figure 7.11: Modular Architecture Example

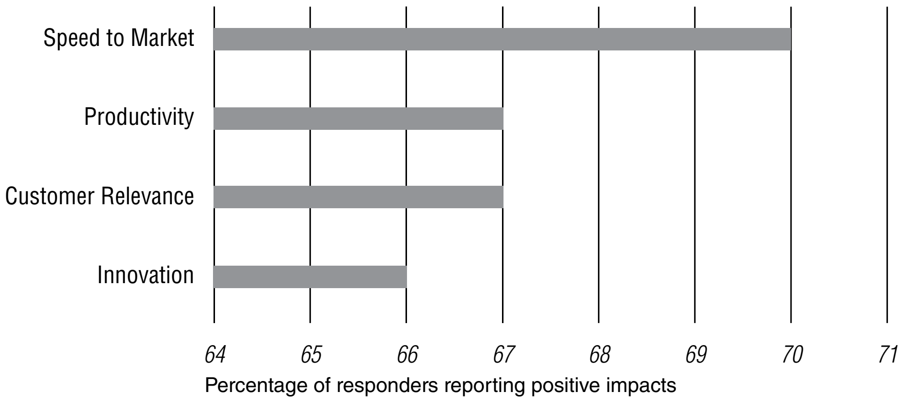


Figure 8.1: Positive Impact of DevOps

Source: Harvard Business Review Analytic Services, Competitive Advantage through DevOps*.*

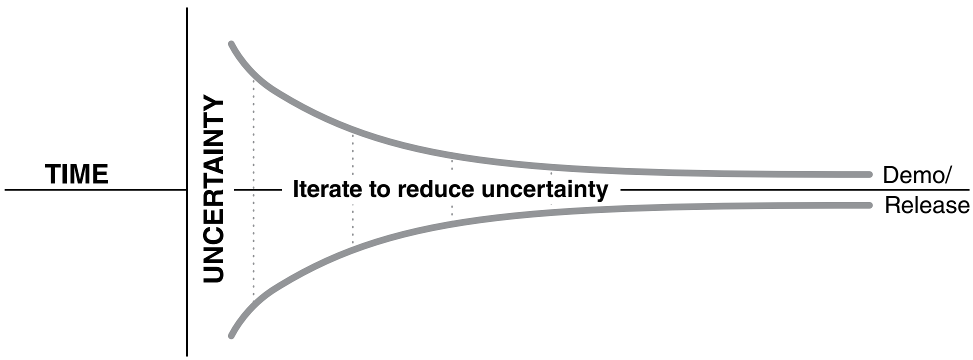


Figure 8.2: Iterate to Reduce Uncertainty

Table 8.1: Queuing Theory Concepts

Term	Definition	Example
Queuing theory	A mathematical study of delays in waiting in line	Tasks queuing up in a computer system
Queuing system	Multiple interconnected queues	Manufacturing systems
Queue	A line of things waiting for processing	A line of people waiting at Starbucks for beverages
WIP	A partially finished product or service awaiting completion.	An individual working multiple activities at one time
Throughput	Average processing rate of the queue	Average number of people serviced at the DMV per hour
Theory of Constraints	Identifies the limiting factor in throughput, frequently referred to as *bottleneck*	People, subject matter experts, machine capacity

Table 8.2: Kanban Practices Defined

Practice	Description
Visualize WIP	The team must be able to visualize the work and the process it goes through.
Limit WIP	The team agrees to limits around how much work can be in process at a given time. This helps decrease flow time.
Manage flow	Use observation and empirical controls to identify and address bottlenecks.
Make policies explicit	The team captures their agreed-upon policies such as WIP limits, definition of *done* at each stage, and other rules for working together. It is the agreement that determines when a work item in the kanban is ready to move from one column (state) to the next column (state).
Implement feedback loops	Obtain feedback from users and stakeholders on work completed
Improve collaboratively, evolve experimentally	The team continuously learns, innovates, and improves the state of the product and the process to improve flow.

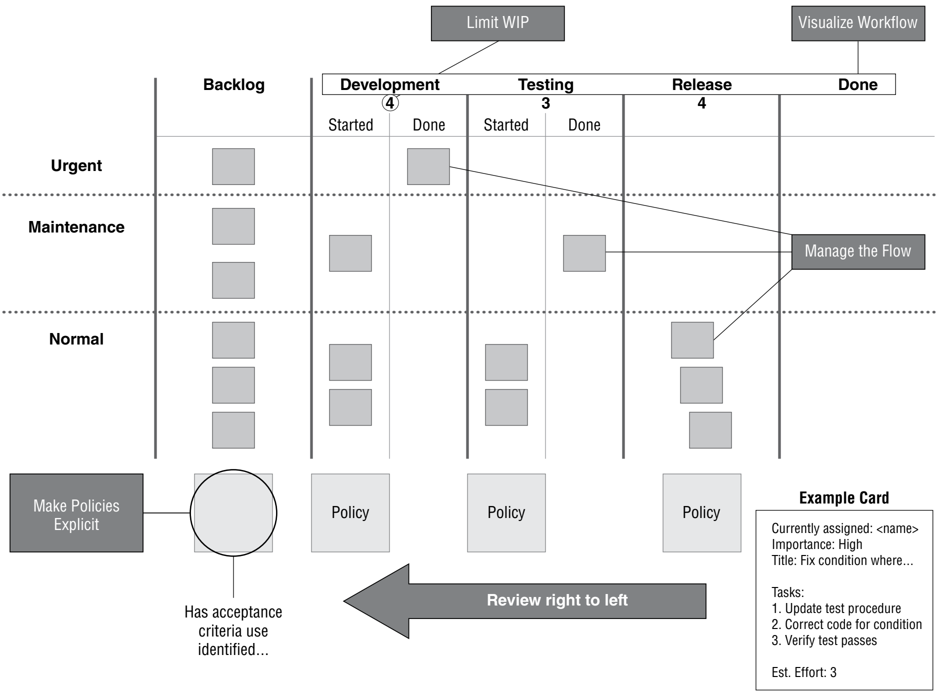


Figure 8.3: Example Kanban Board

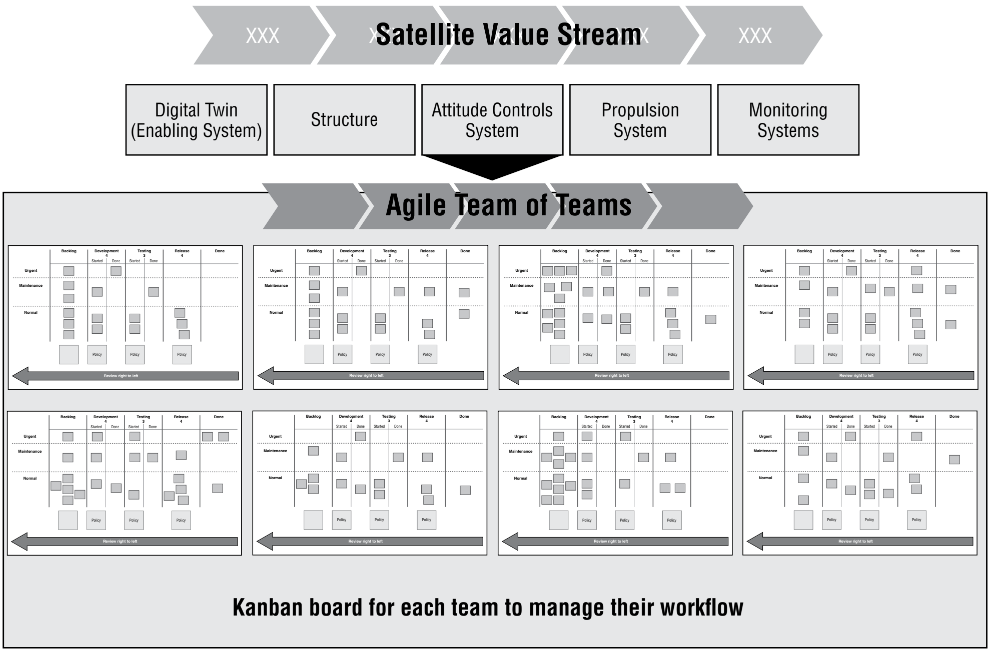


Figure 8.4: Visualizing the Flow of Value through Team Kanban

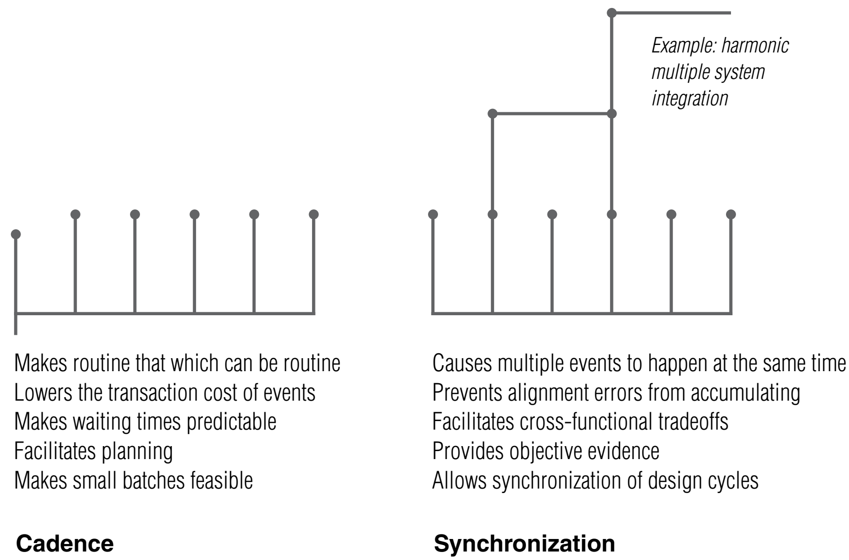


Figure 9.1: Cadence and Synchronization

Source: Josh Atwell et al., *Applied Industrial DevOps*.

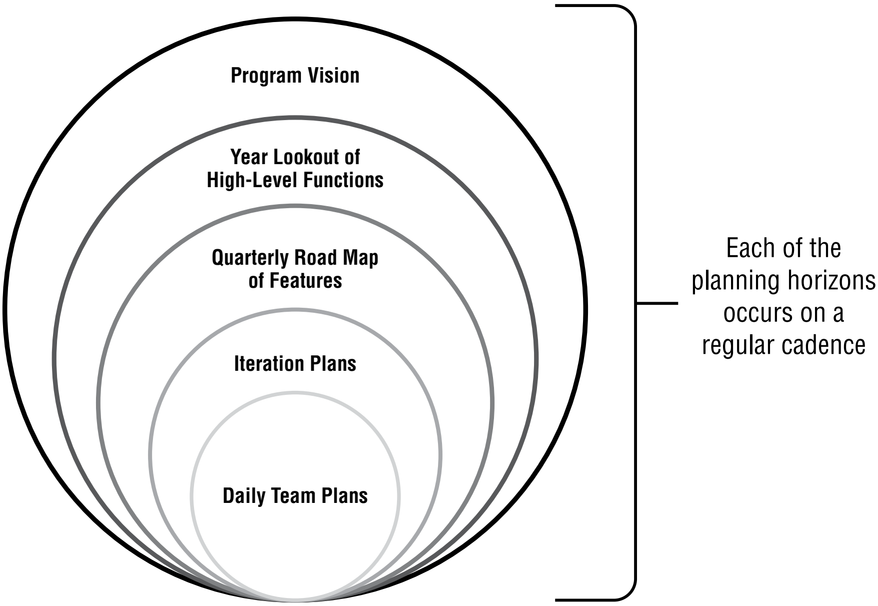


Figure 9.2: Cadence of Multiple Horizons of Planning

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Quarterly planning with teams & customer						
			Iteration planning			
		Integrated demo	Iteration planning			

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		Integrated demo	Iteration planning			

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Feature refinement for next quarter		Integrated demo	Iteration planning			
		Integrated demo				
	Quarterly Planning with teams & customer					
		Iteration planning				

Figure 9.3: Example Schedule Based on the CubeSat Mission

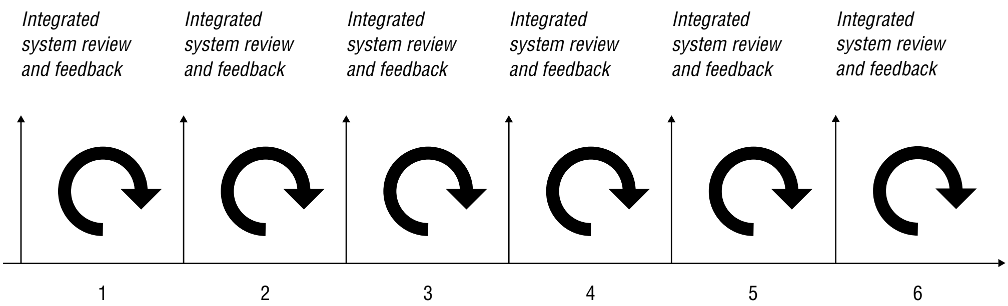


Figure 9.4: Synchronized Cadence

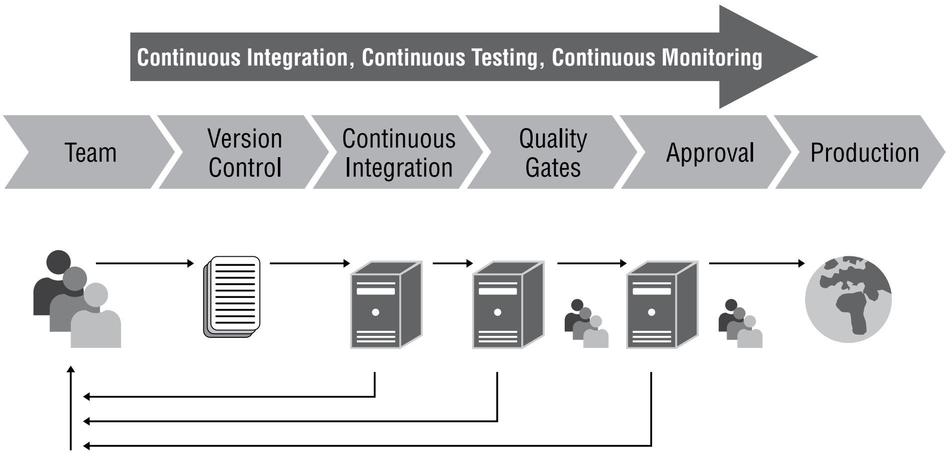


Figure 10.1: CI Pipeline to Optimize the Flow of High-Quality Features to Users

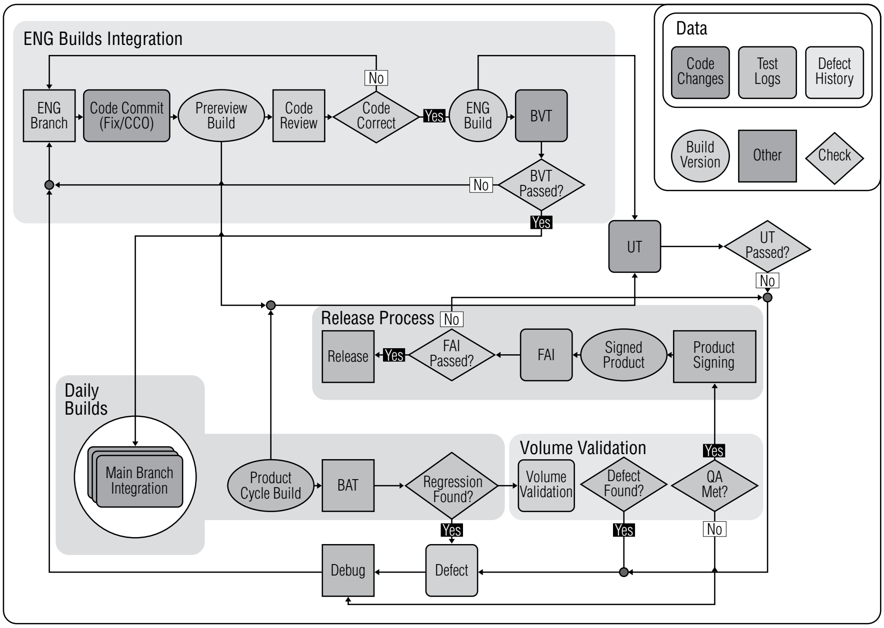


Figure 10.2: CI/CD for Firmware Development for Embedded Systems

Source: Mateusz Kowzan and Patrycja Pietrzak, "Continuous Integration in Validation of Modern, Complex, Embedded Systems."

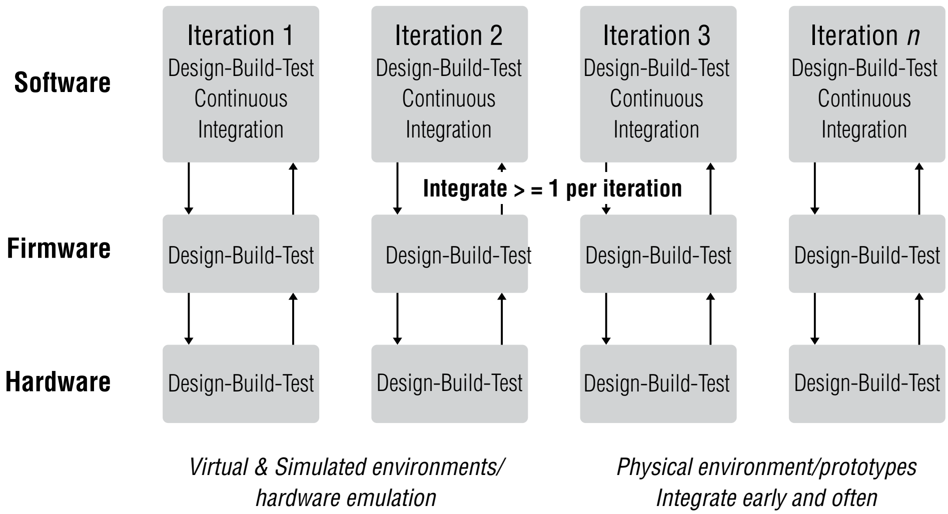


Figure 10.3: Software/Hardware Integration

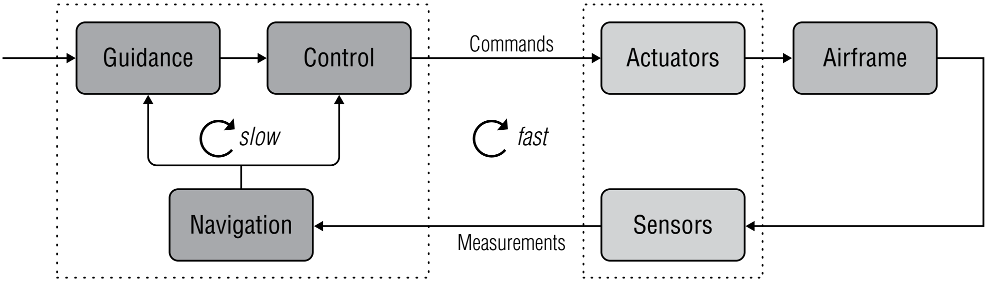


Figure 10.4: Cross-System Integration for Satellite

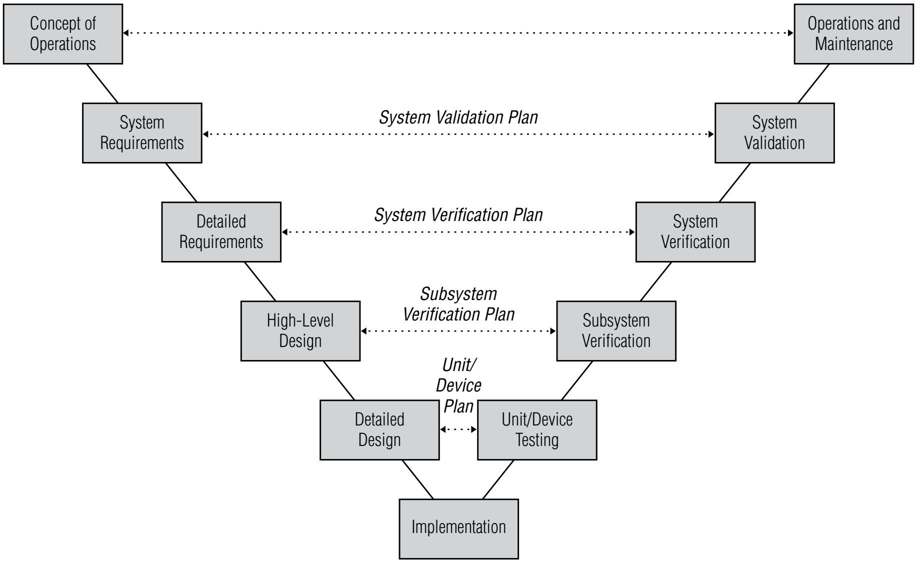


Figure 11.1: Vee Model

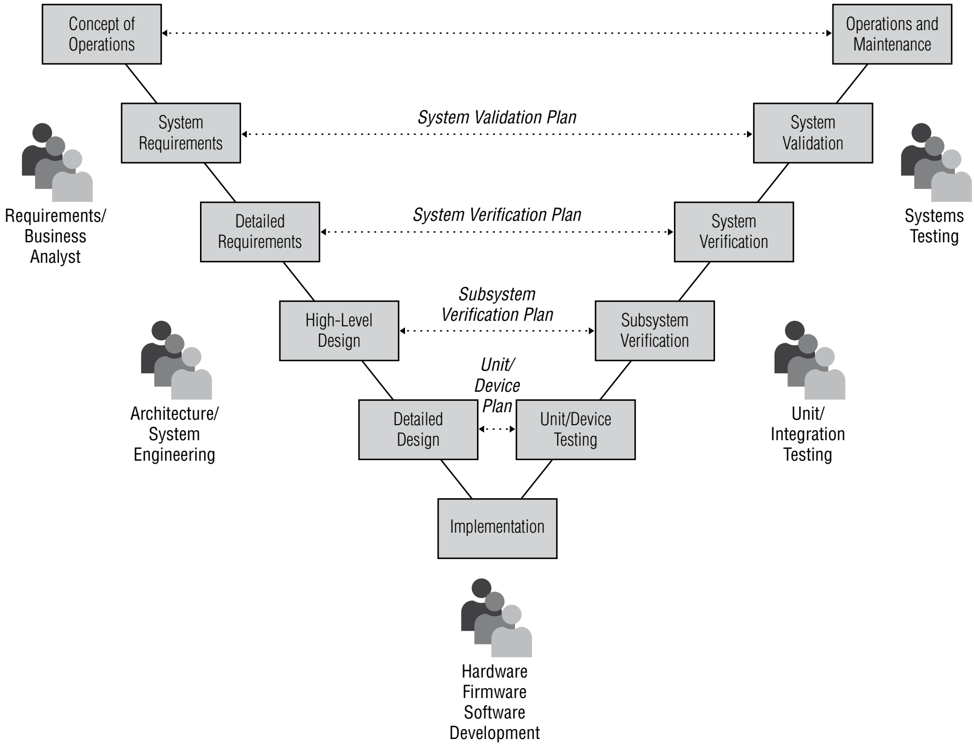


Figure 11.2: Vee Model Teams

Requirement: The system shall be a gray box.

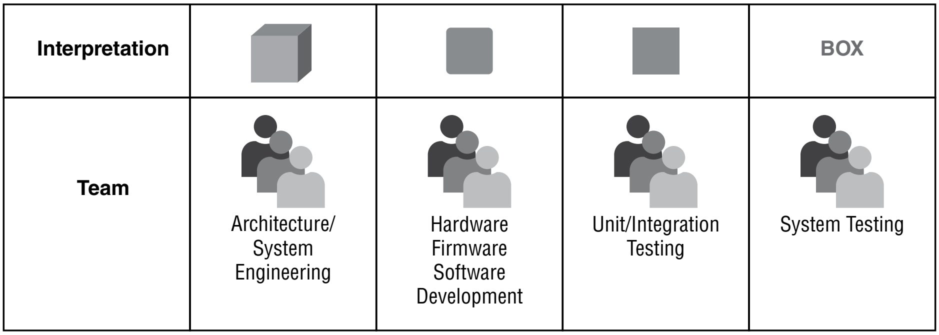


Figure 11.3: Requirements Interpretations

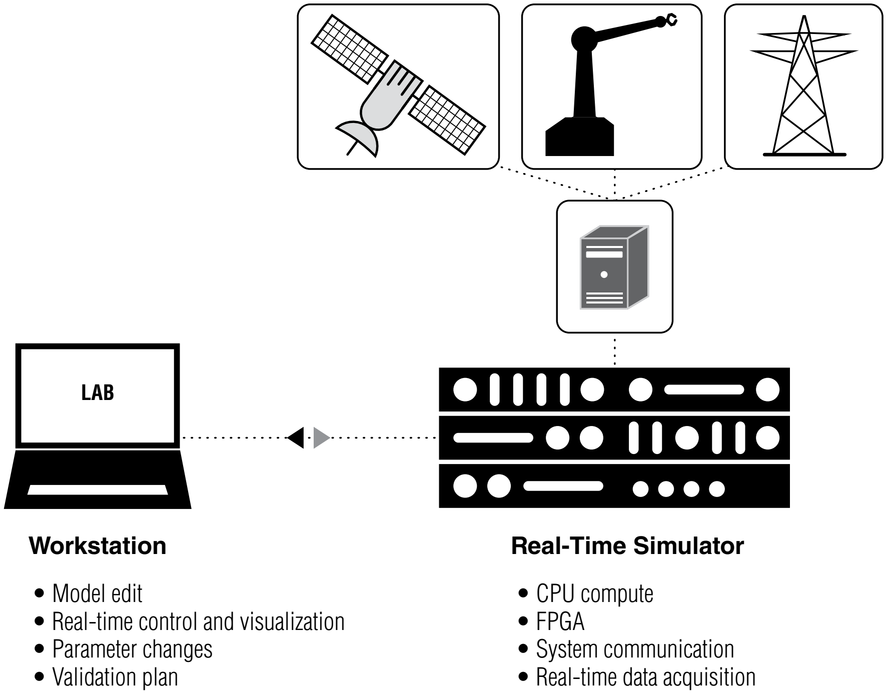


Figure 11.4: Software-in-the-Loop

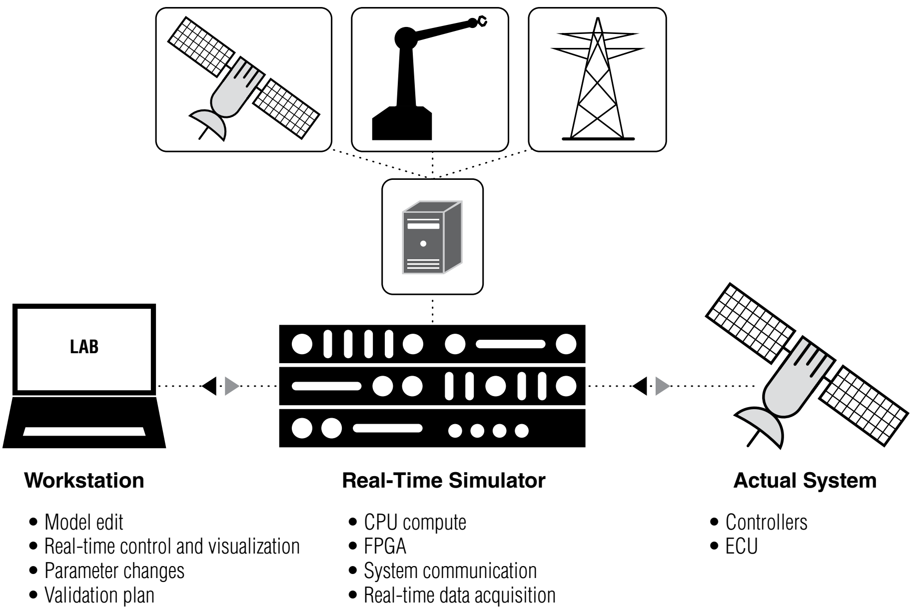


Figure 11.5: Hardware-in-the-Loop

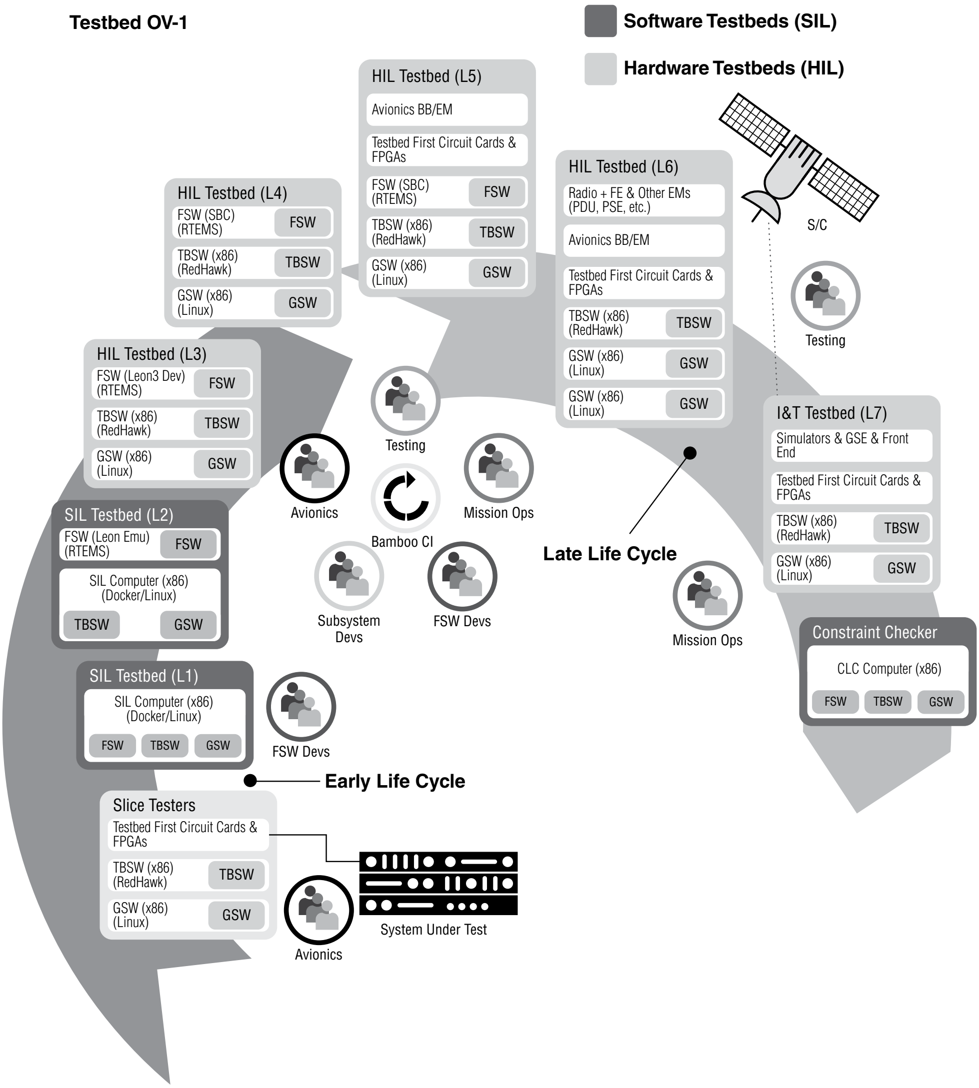


Figure 11.6: The OV-1 of the Johns Hopkins Test Environment

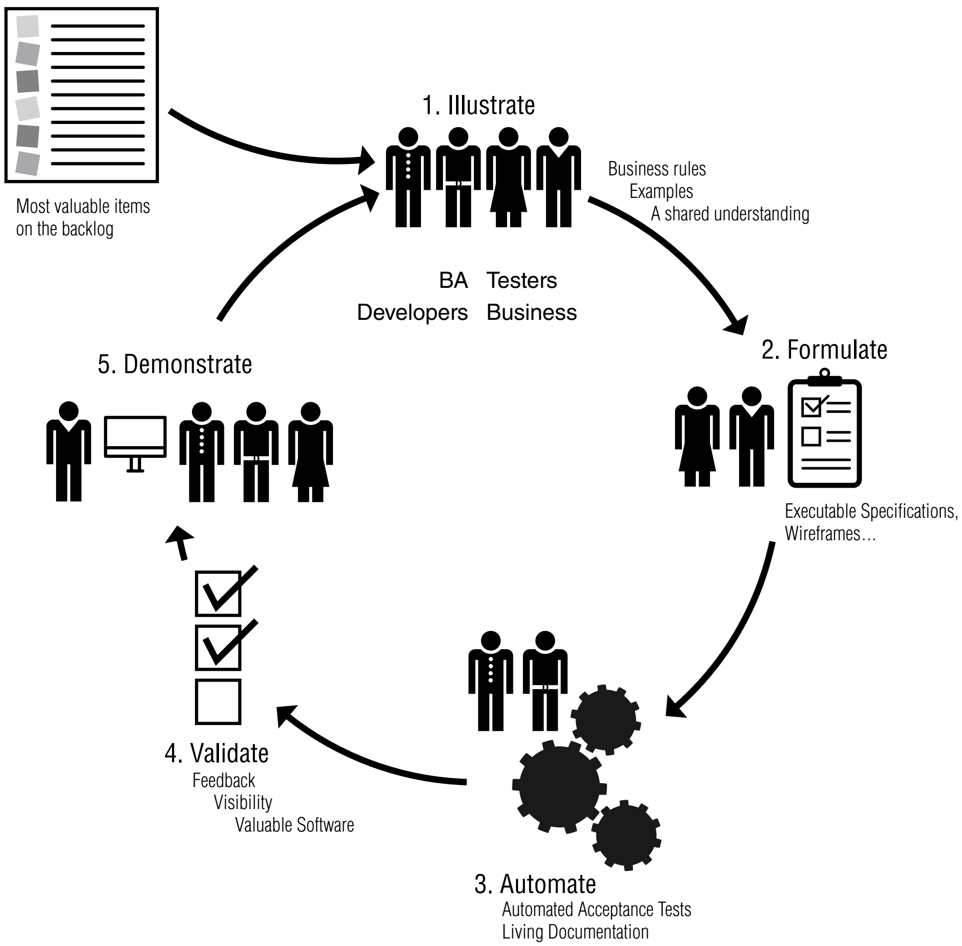


Figure 11.7: Behavior-Driven Development

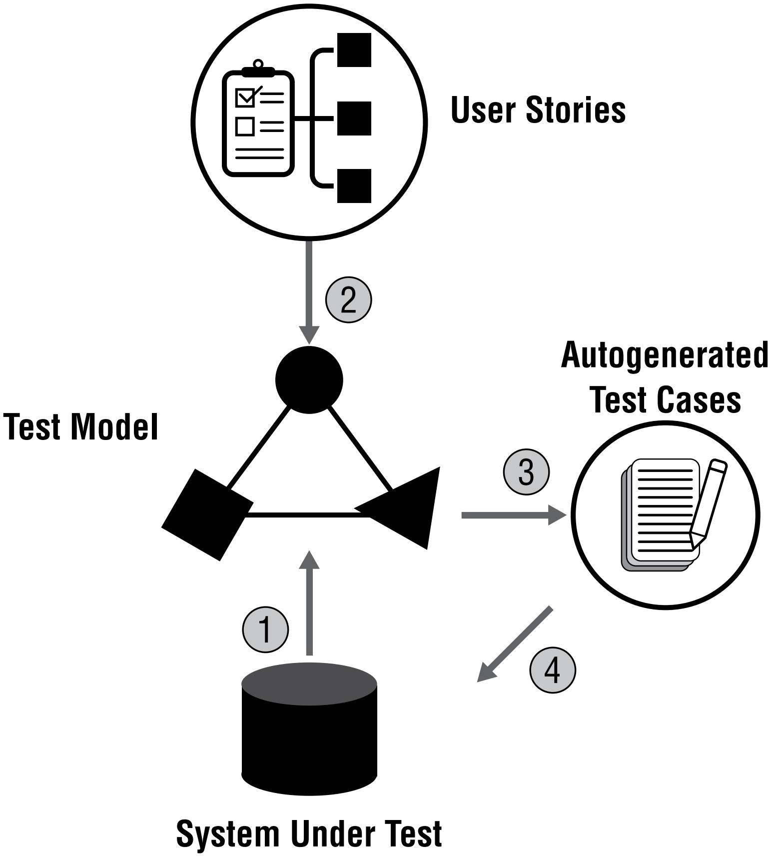


Figure 11.8: Model-Based Testing

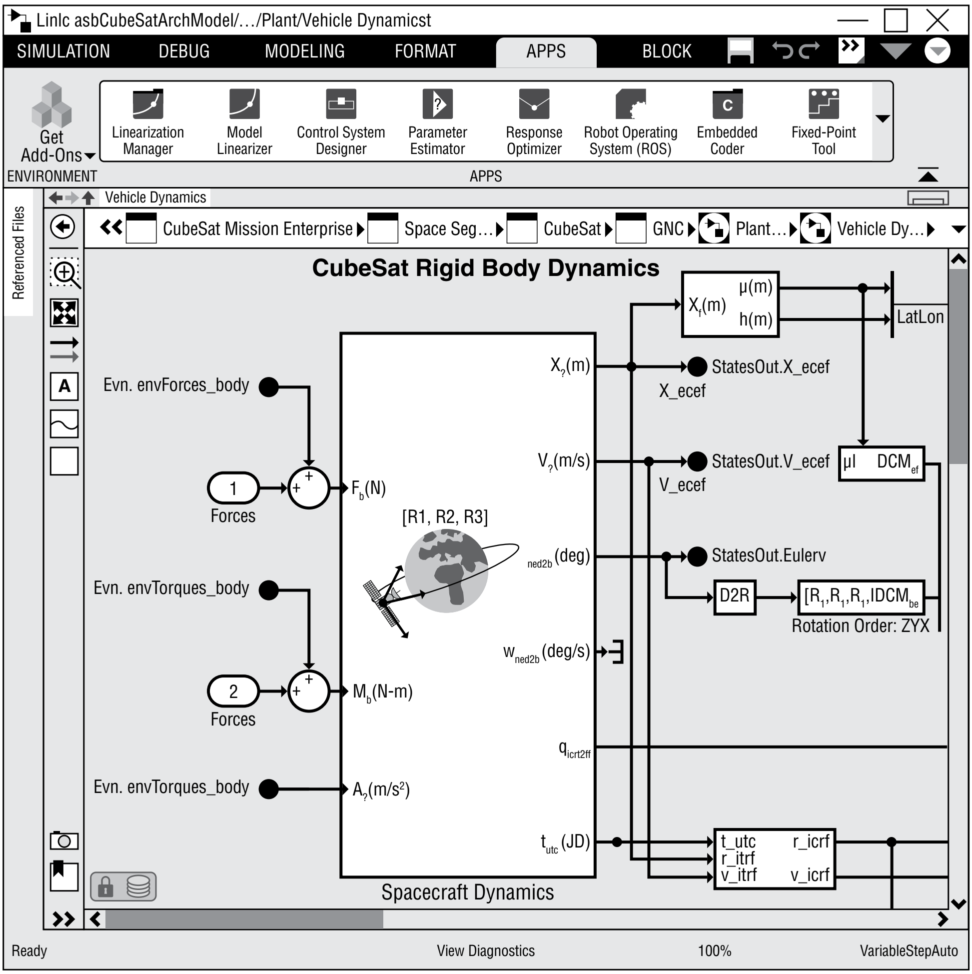


Figure 11.9: MATLAB Model

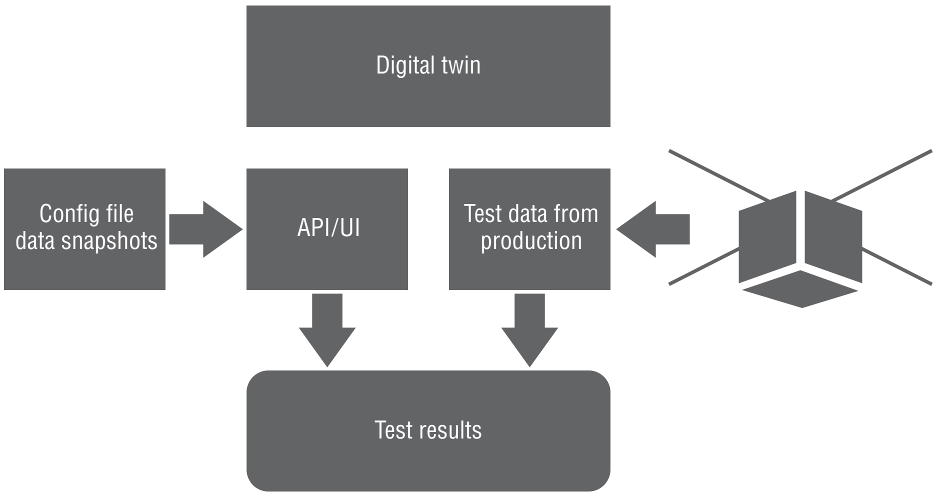


Figure 11.10: Digital Twin–Based Testing

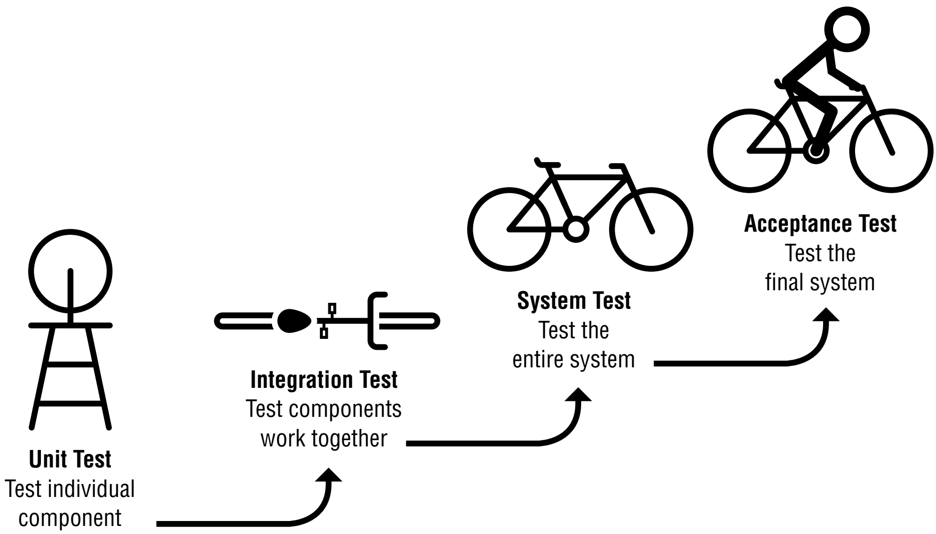


Figure 11.11: Multiple Tiers of Testing

Table 11.1: Facets of System Testing

	Test Type	Description	Example
1	Usability	Testing applied that ensures it is user friendly and easy to use, and that it meets the needs and expectations of the end users.	Validating that the displays on an automobile are easy to read.
2	Functional	Testing focused on verifying the system performs as intended and meets the specified need.	Verifying that a spacecraft can communicate with the ground station.
3	Performance	Testing the system's performance under different workloads and usage scenarios to ensure that it can handle the expected load.	Performing vibration testing to ensure the spacecraft can withstand the vibrations and shock it will experience during launch.
4	Security	Testing that validates the system's security features and ensuring that it is protected against potential security threats and vulnerabilities.	Performing penetration testing that involves simulating an attack on a satellite to determine vulnerability.
5	Compatibility	Testing the system's compatibility with different operating systems, hardware platforms, and other applications.	Testing to validate satellite's communications protocols are compatible with those used by ground systems and they can receive and transmit data

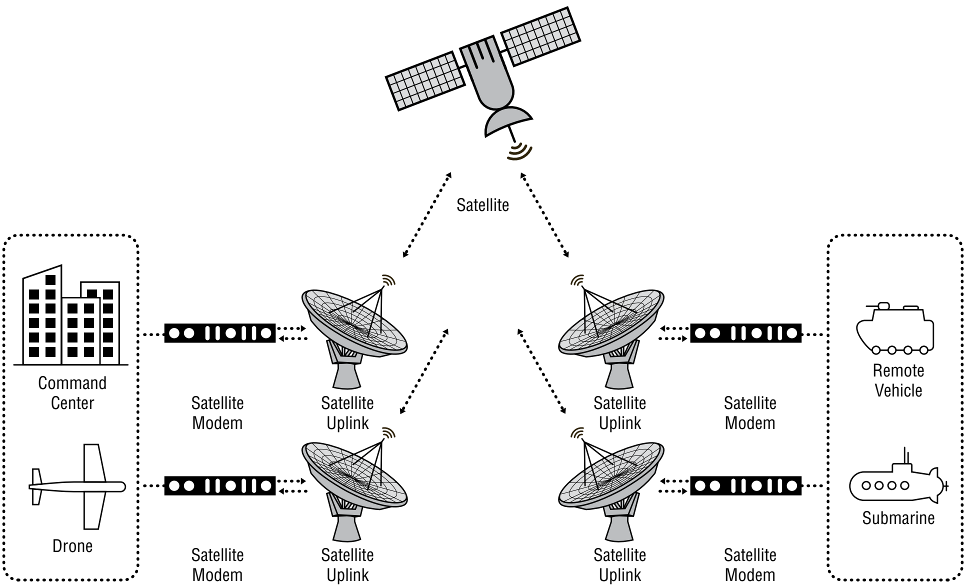


Figure 11.12: System Testing

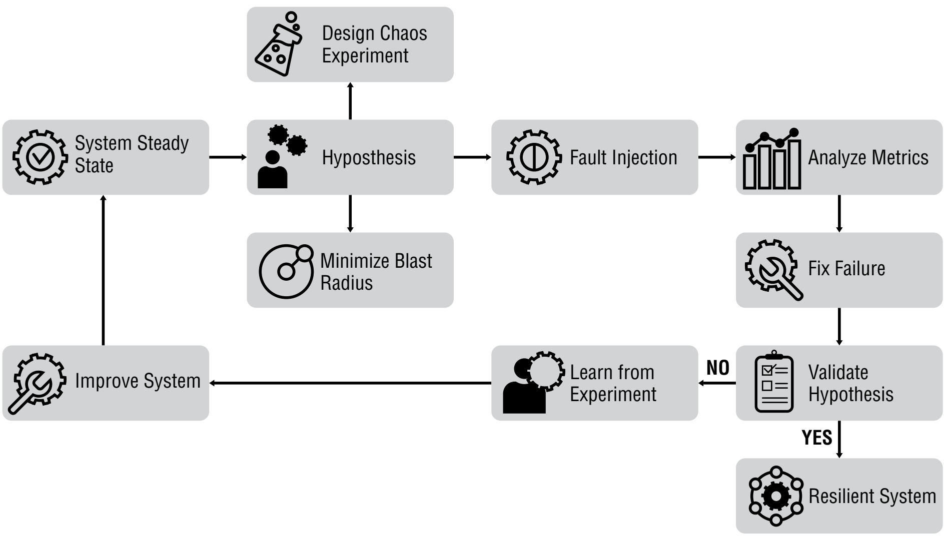


Figure 11.13: Chaos Engineering Explained

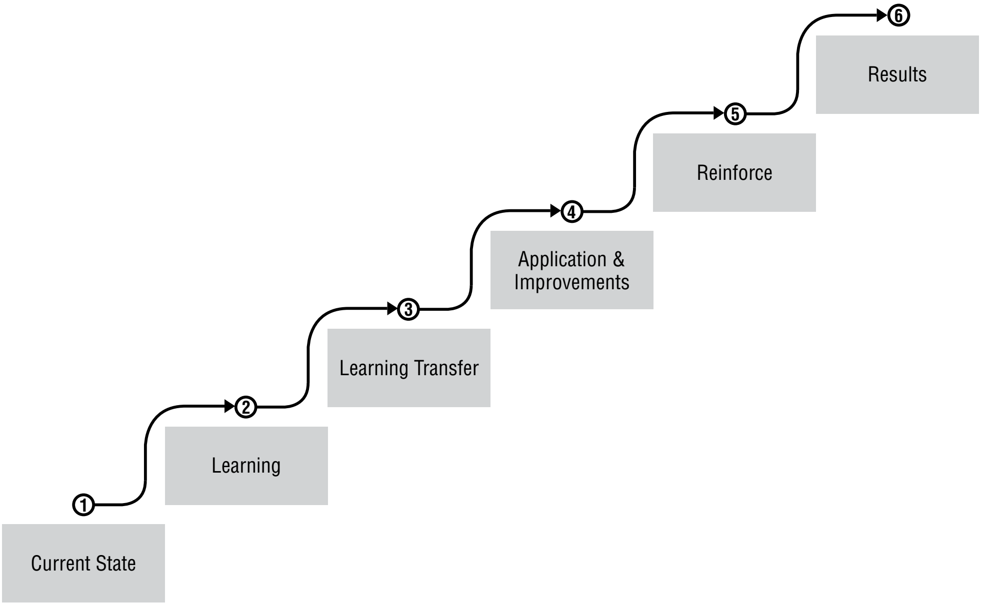


Figure 12.1: Steps of Learning

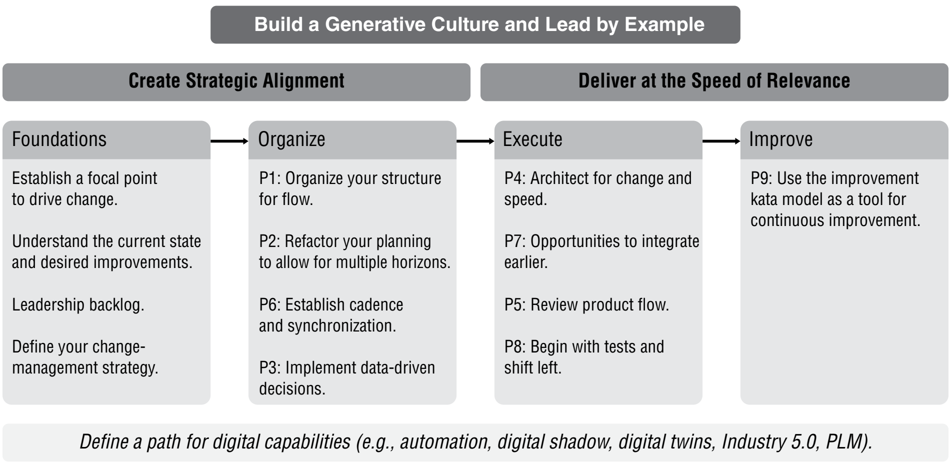


Figure 13.1: Industrial DevOps Framework

Table 13.1: Westrum's Organizational Typology Model

CATEGORY	DESCRIPTION	CHARACTERISTICS
1. Pathological (Power oriented)	Organizations are managed through fear and threats. People are incentivized to hoard or withhold information to improve their power stance.	Low cooperation Messengers shot Responsibilities shirked Bridging discouraged Failure leads to scapegoating Novelty crushed
2. Bureaucratic (Rule oriented)	Organizations protect departments. The members of the department want to lead the organization and follow a strict set of rules where all members are treated equally.	Modest cooperation Messengers neglected Narrow responsibilities Bridging tolerated Failure leads to justice Novelty leads to problems
3. Generative (Performance oriented)	Organizations focus on the mission. The organization implements the mission by intent. Everything is about successfully meeting goals and objectives.	High cooperation Messengers trained Risks shared Bridging encouraged Failure leads to inquiry Novelty implemented

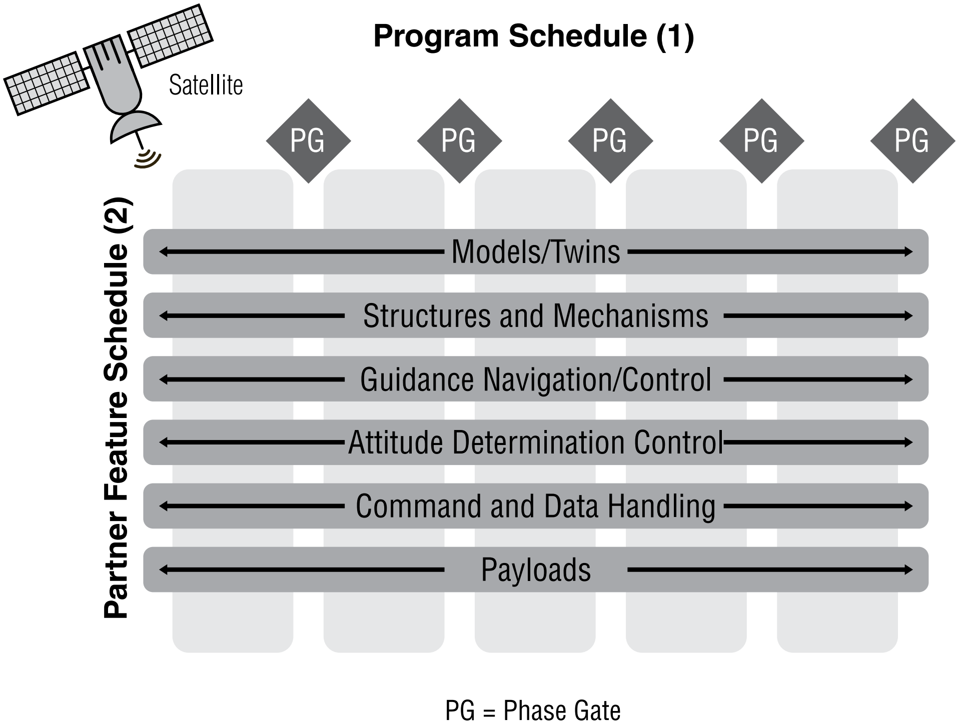


Figure 13.2: Schedule Grid

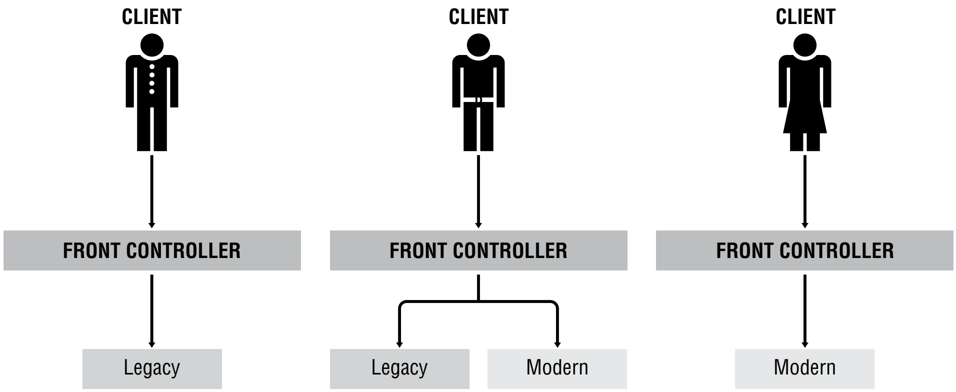


Figure 13.3: Strangler Pattern in Action

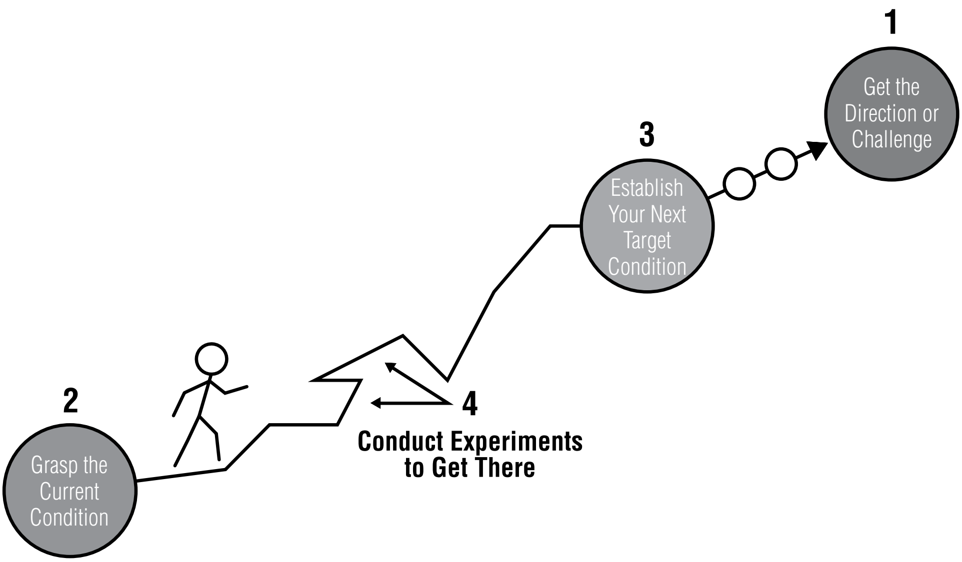


Figure 13.4: The Toyota Improvement Kata

Source: Mike Rother, Toyota Kata*.*

Table 13.2: Principles Focused on Organization and Structure

P1: Organize for the Flow of Value	
Connection to Other Principles	P2: Apply Multiple Horizons of Planning P4: Architect for Change and Speed P7: Integrate Early and Often
How the Principles Work Together	(P2 and P7) Once the value stream is understood, teams are organized around the value stream. The product is defined from vision and high-level yearly plans, with detailed backlog definition happening closer to execution. At the implementation level, teams are cross-functional and develop the product through a series of short iterations with frequent feedback loops to regularly validate and verify the features while continuously integrating and testing throughout the iterative development cycle. (P4) Consider how the system is architected and how the organization aligns the teams. The teams must be organized around the flow of value delivery. Define your value stream and the products that the value stream produces. Organize the people around the flow of value. Revisit the architecture of the system and ensure modularity and reduce dependencies. While the backlog defines new functionality, it must also include the work that needs to be done to continuously evolve the architecture.
P2: Apply Multiple Horizons of Planning	
Connection to Other Principles	P6: Establish Cadence and Synchronization
How the Principles Work Together	Each of the multiple horizons of planning (P2) occurs on a regular cadence. Each horizon of planning yields empirical data demonstrating the success of the plan as it is executed and identifying necessary adjustments. The observe-orient-decide-act (OODA) model is utilized in the military to quickly respond in changing and unpredictable environments. Just as in the OODA model, with each cycle, take what you observe and learn and feed that learning into the next cycle for planning and implementation.

P3: Implement Data-Driven Decisions	
Connection to Other Principles	P2: Apply Multiple Horizons of Planning
How the Principles Work Together	The backlog is defined at multiple levels of decomposition (epic, feature, user story, task) and is planned within multiple horizons of planning. As backlog items get closer to implementation, they are further refined. Each item contains a description, who needs the capability, business benefit, and acceptance criteria. As backlog items are demonstrated, the objective evidence oft what is and is not working is fed into the next planning cycle.
P6: Establish Cadence and Synchronization	
Connection to Other Principles	P7: Integrate Early and Often
How the Principles Work Together	We bring the understanding of cadence and synchronization with us into the cyber-physical world as product teams plan, develop, and deliver system capabilities. Together they define the standard of repeatable planning sessions, large-system integration, and demonstrations of integrated working capabilities. With large cyber-physical solutions, these demonstrations are implemented in a hybrid manner with a mixture of digital and physical artifacts. As your organization defines the cadence and synchronization points, communicate the need for CI at the system level as early as your environment can enable. Teams that integrate and demonstrate functionality together will want to be on the same cadence so their synchronization points align.

Table 13.3: Principles Focused on Execution

P4: Architect for Change and Speed	
Connection to Other Principles	P1: Organize for the Flow of Value
How the Principles Work Together	Intentional modular architecture with standardized interfaces reduces the number of dependencies, which enables the flow of value to stakeholders. The product backlog defines new business-facing functionality as well as enablers to evolve the architecture. Organize teams around the flow of value in concert with your architecture. One approach when considering our CubeSat example is having stream-aligned or complicated subsystem teams implementing capabilities such as attitude control, attitude determination, and attitude estimation and filtering while having a platform team build the infrastructure needed to run the software.
P5: Iterate, Manage Queues, Create Flow	
Connection to Other Principles	P1: Organize for the Flow of Value
How the Principles Work Together	Teams are organized around value streams to improve the flow and delivery of value. This yields the benefits of iterative development and incremental delivery. While iterative development is often thought to be for software development only, this is not the case, as we have conveyed throughout this book. Embedded systems and hardware teams are now engaged in iterative development cycles. Players in the space industry have demonstrated this advantage. For example, Rocket Lab has "demonstrated the ability to support rapid integration and short notice customer-driven changes in launch schedule, inclination, and launch site."[9] As you are getting started, organize your teams around the value stream, create product backlogs that align with your cross-functional Agile team structure, and set up the tool environment that enables iterative development across software and hardware with feedback loops in manufacturing and with the customer.

P7: Integrate Early and Often	
Connection to Other Principles	P3: Implement Data-Driven Decisions P5: Iterate, Manage Queues, Create Flow
How the Principles Work Together	The goal of integration is to ensure that all of the systems being developed can communicate and share data effectively. Integrating early and often is not only from a software perspective; it means integrating at the system level. Integrating early and often not only buys down risks while ensuring fit for purpose, but when coupled with iterative demonstrations, it provides real-time information and data about what is working or not working. Metrics are reviewed to understand the current system state. As we manage and improve flow, integrated tool sets and dashboards generate these metrics, which can be used to see where the bottlenecks are in the system. Build the system iteratively, integrate early and often, improve flow, and use data to understand the current state and to make decisions on the next steps to take in developing the solution or improving the flow.
P8: Shift Left	
Connection to Other Principles	P3: Implement Data-Driven Decisions P5: Iterate, Manage Queues, Create Flow
How the Principles Work Together	Through experience, we have found that reviewing requirements through a lens of how they will be validated and verified has shown to ensure executable requirements. As you get started with a shift-left implementation, define your test strategy to incorporate a shift-left testing mindset. Acceptance tests are written before development begins. Shift-left manufacturing creates regular feedback to optimize test processes by identifying areas that are time-consuming or difficult to test. Verification of designs happens regularly. Through experience, we have found that reviewing requirements through a lens of how they will be validated and verified has shown to ensure executable requirements. As you get started with a shift-left implementation, define your test strategy that incorporates a shift-left testing mindset. Shift-left manufacturing creates regular feedback to optimize test processes by identifying areas that are time-consuming or difficult to test. Verification of designs happens regularly.

Table 13.4: Principles Focused on Continuous Improvement

P9: Apply a Growth Mindset	
Connection to Other Principles	All principles
How the Principles Work Together	A growth mindset requires continuous learning and relentless improvement. Organizations should continuously measure flow and reduce bottlenecks. Engage the customer regularly and gather feedback to relentlessly improve the product and process. Employ retrospectives at all levels and have a defined process for prioritizing new backlog items created from the retrospectives. When there is a systemic problem, take action to determine root causes and the next opportunities. These improvements should be made visible to the organization through dashboards and events. Never stop learning. Embrace change.

Table 14.1: Barriers and Challenges to Industrial DevOps Adoption

List of Barriers and Challenges to Industrial DevOps Adoption
Lack of consistent implementation of Industrial DevOps–related processes and practices.
Lack of a growth mindset and hesitancy to rethink current ways of working.
Lack of skills and experience hinders the scaling of Industrial DevOps.
Lack of engagement and organizational alignment leaves people unmoved.
Challenges with complexity and dependencies and scaling across teams of Agile teams.
Challenges with regulated environments.

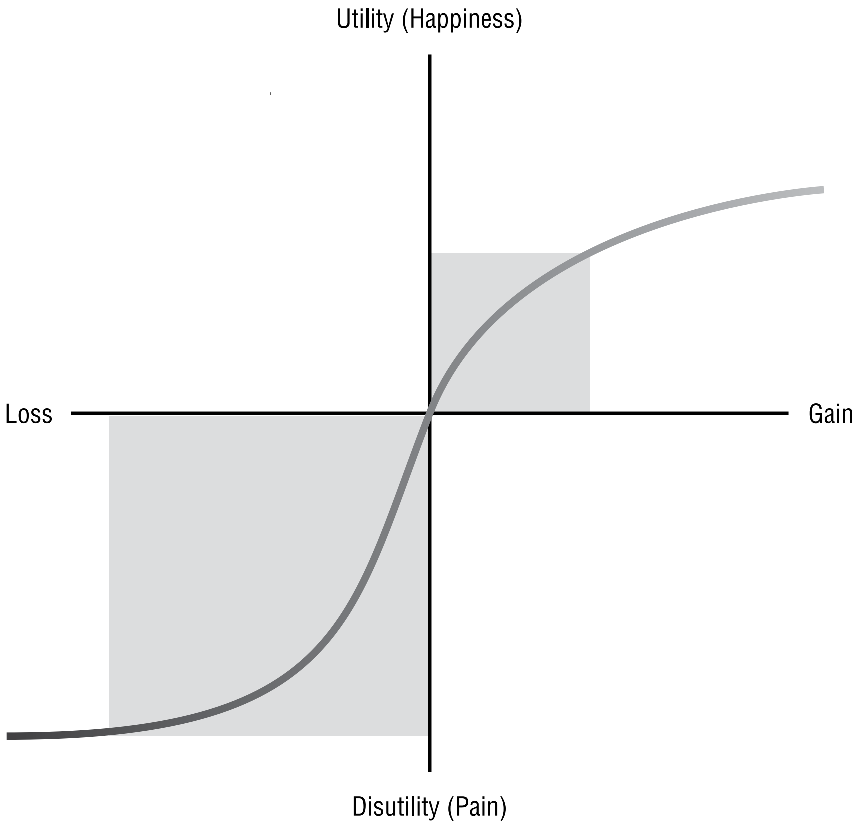


Figure 14.1: Prospect Theory

Source: Charlotte Nickerson, "Prospect Theory in Psychology."

Conclusion

> Change is a common thread that runs through all businesses regardless of size, industry and age. Our world is changing fast and, as such, organizations must change quickly too . . .
>
> —Kurt Lewin, Change Management

Our path to Industrial DevOps started twenty years ago. Over the years, we have worked in a variety of settings for different customers across a myriad of products and integrated systems. We have experienced successes. We have struggled. We have continued to learn along the way. Based on this journey, we have collected and shared with you success patterns that define the Industrial DevOps principles. As you embark on your Industrial DevOps journey, there are three critical insights to guide your journey:

1. Industrial DevOps applies to the entire system.
2. Digital capabilities enable fast feedback loops and shift-left practices.
3. People and culture are the key to success.

Industrial DevOps Applies to the Entire System

Industrial DevOps applies to the entire system (the organization and the cyber-physical system). Thus, it is essential to apply systems thinking and look holistically across the system for improvements and opportunities. This has been a primary lesson for us and continues to be validated as we exchange with companies from around the world who build cyber-physical systems. While Lean, Agile, and DevOps have existed for decades, it is the weaving together of these practices across the value stream that improves the flow and delivery of value. With the rapid advancement of digital capabilities and tooling, now is the opportune time to embrace Industrial DevOps.

As you begin your Industrial DevOps journey, remember to start with "why." Why are you making this shift? What is the imperative for your organization that makes now the right time? According to Simon Sinek in *The Infinite Game*, for organizations to stay in the game, it is no longer about "who wins or who's the best" but rather about building organizations that are healthy and able to survive for decades to come.[1] Work across the organization to define the business outcomes you are after, define your future state, and create your road map for getting there. You will need to adjust along the way—not because your initial plan was wrong but because the ecosystem around you continuously evolves. Industrial DevOps principles enable you to inspect, adapt, and course correct so you can take advantage of emerging markets, new technologies, and changing priorities of the business. Take with you the coaching tips that we have provided with each chapter and use them as a guide to take your next steps. Embrace a growth mindset. Commit to continuous learning.

Digital Capabilities Enable Fast Feedback Loops and Shift-Left Practices

The advancement of digital capabilities and technologies has created an environment in which we can now apply what we have learned from the decades of software development to the world of hardware and manufacturing. Digital capabilities have enabled shift-left practices in which testing, compliance, and design for manufacturing are now part of the integrated and iterative development process. This shift results in higher-quality products and reduces rework downstream by building in quality and inspections along the way through integrated system demonstrations. Teams' processes enable short feedback cycles in which they get feedback on cadence from other teams and stakeholders such as customers, business leads, and users and can quickly adjust based on the feedback received. Status is not focused so much on task completion, as now it focuses on demonstrations based on defined acceptance criteria.

With physical hardware, we are able to shift left as we move physical development and testing into digital environments. The development of physical systems continues to take advantage of emulators, simulators, and prototypes along with the advancement of emerging capabilities such as 3D printing, additive manufacturing, digital threads, digital twins, and virtual reality and augmented reality (VR/AR). Digital capabilities impact not only how we develop cyber-physical systems but also the factory used to build the system. The emergence of Industry 5.0 and the smart factory brings in a wide array of digital capabilities. New capabilities are tested early, and with improved safety, through the use of automation, robotics and autonomous systems, VR/AR, AI, and more. The range of possibilities to improve operational efficiencies and deliver faster continues to emerge. Shifting left enables early validation of designs, reduces risks, and builds in quality to build systems better and faster.

People and Culture Are the Key to Success

Who builds these better systems faster? It is the people. Therefore, the people and culture of the organization are the key to success. Meet people where they are in their Industrial DevOps journey and take time to understand their existing culture. In many instances, you will discover that it is not that people aren't willing to adopt new ways of working. The hesitancy is more often because they don't understand the reason for the change, they don't know what it is they are being asked to do, there is fear of failure, and they are not given the time and support they need to learn. By creating a culture of continuous learning, you can begin to address these concerns. Listen to what they need and provide them with the resources, training, digital tools, and environment necessary to do their job.

With the complexity of large cyber-physical systems, it is recommended that adoption of Industrial DevOps principles starts with one of the smaller value streams of the system and then grows. Build off the smaller successes to generate more success. Our experiences align with Jonathan Smart's thinking as described in *Sooner Safer Happier*:

> . . . people have a limited velocity to unlearn and relearn. You cannot force the pace of change, even if you think that you are. The outcome will be new labels on existing behaviors, the robotic maneuvers of Agile, people in an agentic state waiting for the next order, and little actual agility. . . . *apply an agile approach to agility and achieve big through small.*[2]

Where to Now?

Digital capabilities and new technologies are growing at an unprecedented rate. As we write this book, we continue to learn more every day about advancements around us. The world of artificial intelligence (AI) has picked up speed, with the impact still largely unknown. Based on current observations, it will change how we build systems, from development through production and delivery. It will change how we learn. Rote tasks will become more automated. Higher-level thinking skills and creativity of our teams will become increasingly important. For instance, a developer may use AI to help design test cases or to find errors, and AI will learn to ensure those errors are not repeated.

However, the developer needs to know what questions to ask and what processes need to be followed and to be able to validate the accuracy of the information received. And this is just a small example of the changes swarming upon us. Machine learning, AI, autonomous systems, robotics, continued enhancement of VR/AR, and more are evolving each day. Continuous learning and adapting to changing environments is required for survival and will provide opportunities that we have yet to explore.

We know this journey is not finished, because the world has not stopped evolving. The "digital age" has only just begun.

The Next Step Is Yours

We have shared and demonstrated the Industrial DevOps principles throughout this book and have described various tools and techniques you can use to help you. We have provided examples of companies demonstrating these principles, and you can see the world around you changing. Reflecting on what you have read, define your next steps in adopting Industrial DevOps principles to build better systems faster.

As we said at the beginning of this book, "Companies that solve this problem first will increase transparency, reduce cycle time, increase value for money, and innovate faster. **They will build better systems faster and become the ultimate economic and value delivery winners in the marketplace.**"

The next step is yours.

Thank you for being part of our learning journey.

APPENDIX A

CUBESAT 101

In order to build a common mental model while discussing the Industrial DevOps principles in this book, we frequently use the example of a CubeSat (a cube-shaped miniature satellite). We use this example because they are cyber-physical systems that are relatively simple to understand. If you're not familiar with CubeSats, this section will break the basics down for you.

Machine-based satellites are used for a wide variety of purposes to do everything from weather forecasting to using a global positioning system (GPS). Many of us make use of applications such as Google Maps in our everyday lives to obtain directions to go from one place to another.

Traditional satellite development has a high barrier to entry due to the expense of design and launch. According to GlobalCom, a typical weather satellite costs about $290 million to build and between $50 and $400 million to launch into orbit.[1] In the late 1990s, two professors from California Polytechnic State and Stanford wanted to help their students gain hands-on experience with engineering satellites. They introduced what is now referred to as CubeSats, which are miniature satellites that are also cost-effective. A typical CubeSat is a ten-centimeter, or four-inch, cube with a mass of less than 1.5 kilograms (three pounds).

CubeSats are small but mighty cyber-physical systems (see Figure A.1), which is why we selected them for our example. They can be as simple or complex as we want to make them.

Interestingly enough, the use of CubeSats has exploded over the last fifteen years, with a target demand expected to reach $857.39 million by 2030.[2] They have primarily been flown in low Earth orbit (LEO), but now CubeSats are moving out to deep space. You can build a CubeSat for $1,000 with a launch price of between $10,000 and $40,000. This has changed the game. Because of their size, they are exponentially cheaper to launch. And their defined technical standards make it easier for new players to enter the market. They can be grouped together to create larger capabilities, such as Starlink, a satellite internet constellation operated by SpaceX.

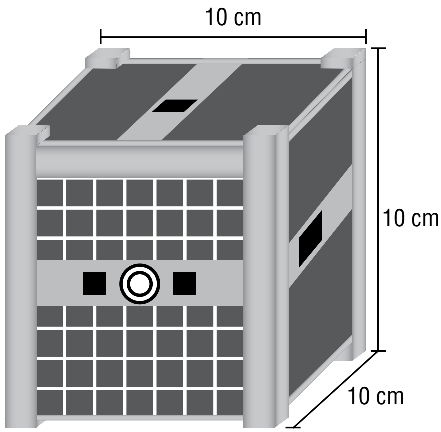


Figure A.1: CubeSat Dimensions

CubeSat Architecture

We are going to share a high-level architecture and design of CubeSat to make our examples throughout the book easier to follow. Our CubeSat components are illustrated in Figure A.2.

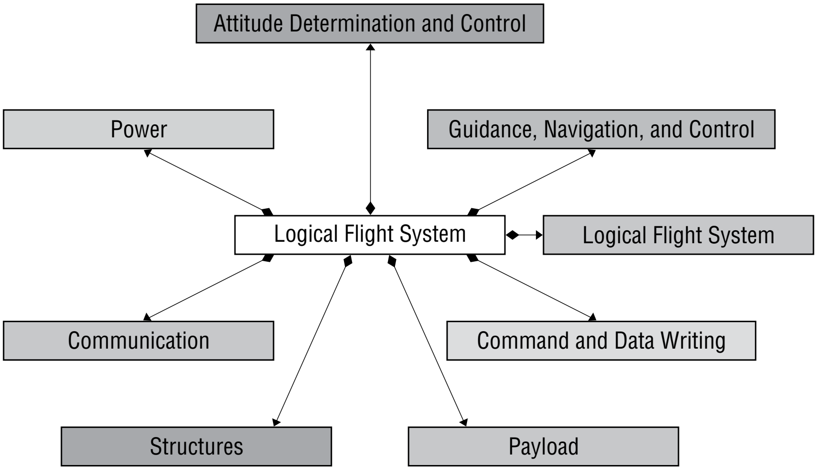


Figure A.2: CubeSat Logical Component Architecture

The description of each of the components is outlined in Table A.1.

Table A.1: CubeSat Logical Component Description

	Component	Description
1	Attitude Determination and Control	Detect and control orientation of CubeSat.
2	Guidance, Navigation, and Control	Navigation, which tracks current location; guidance, which uses navigation data and target information to determine where to go; and control, which accepts guidance commands.
3	Thermal Determination and Control	Sensors to detect and control temperature of CubeSat.
4	Command and Data Handling	Accept commands from ground and dispatch to the CubeSat.
5	Payload	Collect mission-specific data (weather, location).
6	Structure	Controls for the physical system CubeSat.
7	Communication	Transmit data between CubeSat and ground station on CubeSat flight data and mission data.
8	Power	Collect, store, and regulate energy.

Just as with traditional satellites, CubeSats still have to meet stringent requirements, such as using components able to withstand space conditions. The hardware components for our CubeSat are outlined in Figure A.3.

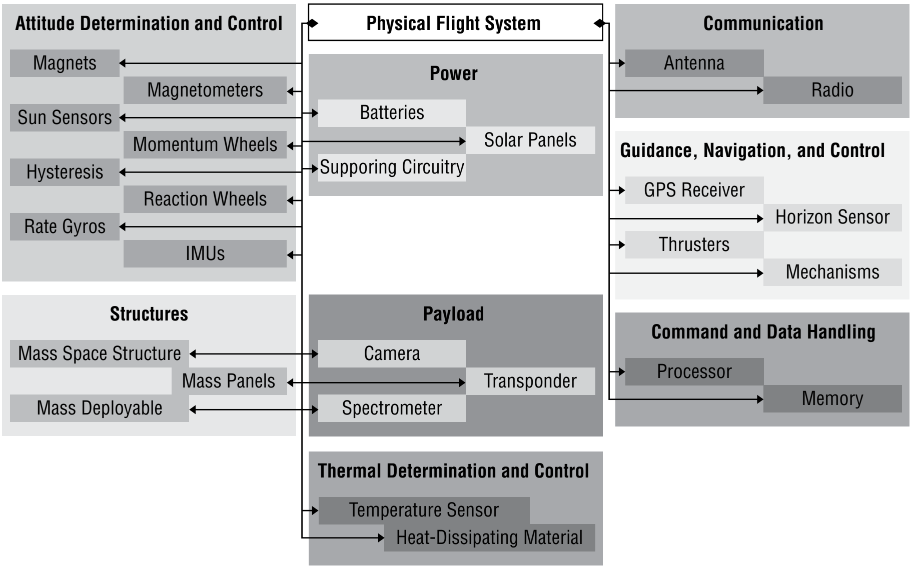


Figure A.3: CubeSat Hardware Component Architecture

In addition to the hardware architecture, we have outlined the software architecture for our simple CubeSat in Figure A.4.

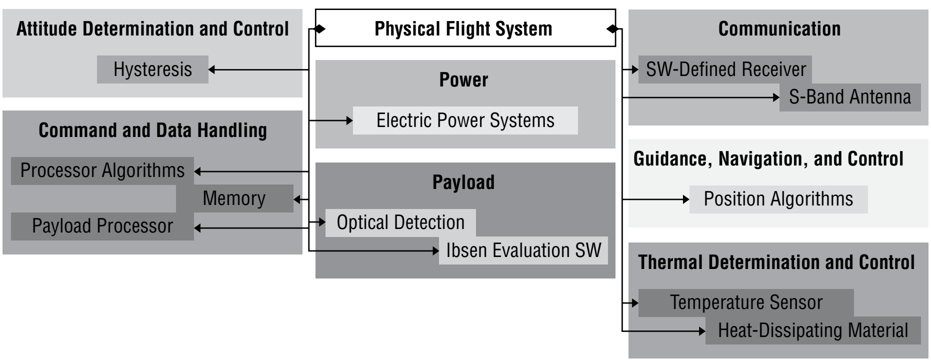


Figure A.4: CubeSat Software Component Architecture

The full physical system of interest is outlined in Table A.2.

Table A.2: CubeSat Physical Components

Subsystem	Component	Description	What It Does
Structure	Access Port	Physical surface interface to CubeSat when it is in the dispenser	Access your CubeSat with RBF pin
	Frame	Physical structure of CubeSat	Hold all of the CubeSat components
	Side Panel(s)	Physical structure of CubeSat	Provide access to components in CubeSat
Power and Thermal	Solar Sensor	Small, lightweight digital sensor that detects UV and infrared light	Determines spacecraft body angles with respect to the sun
	Solar Panel	Physical aluminum panels that collect sunlight and convert into electrical energy	Recharge the batteries in the EPS
	Electron Canon	Used to oust excess electrons	Supports solar wind sail
	Electromagnet	Creates a magnetic field that controls the amount of electric current	Rotates spacecraft by controlling power
	Electrical Solar Wind Sail	Propellantless propulsion system	Moves the CubeSat
	Electrical Power System (EPS)	Contains batteries that power the CubeSat	Powers the CubeSat
	Tether Wheel Motor and Electronics	A spool with long cables that are used for propulsion, momentum exchange, stabilization, or attitude control	Creates momentum to move spacecraft

Table A.2: CubeSat Physical Components Cont.

Subsystem	Component	Description	What It Does
	Tether Endmass	The endmass for the cables	Attaches to tether
ADCS	Command and Data-Handling System	Electronic circuit board that consists of an onboard computer with interface to all subsystems.	The brain for satellites, which handles all operations.
CDHS	Command and Data-Handling System	Electronic circuit board that consists of an onboard computer with interface to all subsystems.	The brain for satellites, which handles all operations.
Communication	Antenna	Copper adhesive tape	Transmit and receive signals for communication
	Radio	Electrical circuit board that transmits and receives UHF/VHF signals	Transmit and receive radio signals for communication
	Communication System	Electronics circuit board containing radio, transceiver, and beacon transmitter and radio	Communicates with controllers on ground or other CubeSats in space
Payload	Camera	Optical device to capture images	Obtains images

CubeSats are advancing space research across a variety of areas such as data transmission (internet accessibility), reduction of orbital debris, science and exploration, earth observation (such as predicting natural disasters), and more.

CubeSat Mission Scenario

CubeSats fulfill many missions. These might include space imagery to understand distances between various objects in space, space weather patterns, imagery and data collection such as tracking of endangered animals, weather events, physical environments, and a variety of science experiments by researchers.

The CubeSat mission we use in this book is to improve weather forecasting accuracy to improve the safety of life and property. In our scenario, we have a targeted customer who is interested in this data to improve their predictive models and provide better weather forecasting. The initial goal of the business is to iteratively launch two hundred CubeSats within the first fifteen months and reduce the lead time to twelve months. The CubeSats need to be replenished every twelve months while building enhanced capabilities. The initial launch has a limited set of capabilities, which will continue to be enhanced with each launch. We include enabling capabilities such as a digital twin instance for each satellite and artificial intelligence as part of the satellite network for advanced communications and data analysis. Due to the number of CubeSats in production and the advanced technologies and growth of the organization, we have eight small teams working on the attitude control systems, which includes the configuration, modeling, and digital twin capability. The business intends to use this mission as a starting point, with plans to extend the business over the next three years.

We selected this mission because it is understandable, regardless of one's background and experiences. The affordability of CubeSats is impacting the space industry, breaking down barriers to entry for small businesses, increasing the ability to innovate and experiment, and improving how we educate and grow the future workforce.

APPENDIX B

INDUSTRIAL DEVOPS BODIES OF KNOWLEDGE

Industrial DevOps is built upon several existing bodies of knowledge. These bodies of knowledge have been proven and validated for decades and are foundational to delivering cyber-physical systems. The bodies of knowledge include Lean, Lean Startup, Agile, DevOps, design thinking, systems engineering and model-based engineering, architecture, and systems thinking. A timeline of these bodies of knowledge is presented in Figure B.1.

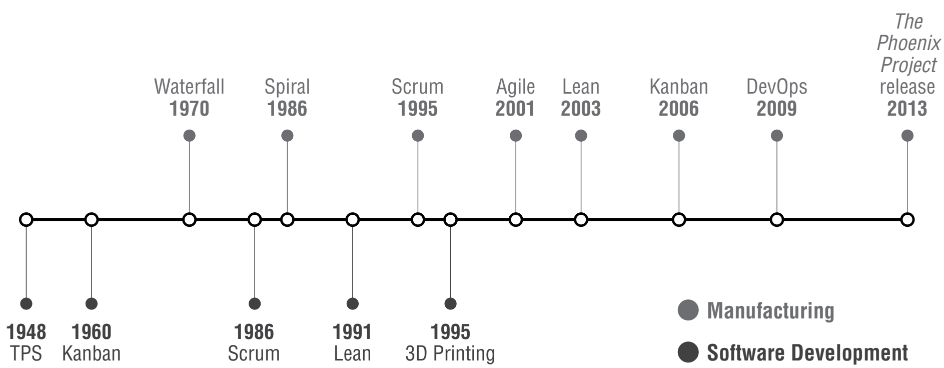


Figure B.1: History of Industrial DevOps Bodies of Knowledge

This appendix will introduce you to the multiple bodies of knowledge we pulled from in defining Industrial DevOps principles for cyber-physical systems. Please note that we are not advocating for one body of knowledge, framework, or method over the others. We suggest understanding all of the tools available in your toolbox and using the correct tools for the problem that you are trying to solve. In most cases, you will find a composite of multiple tools is the right answer (see Figure B.2).

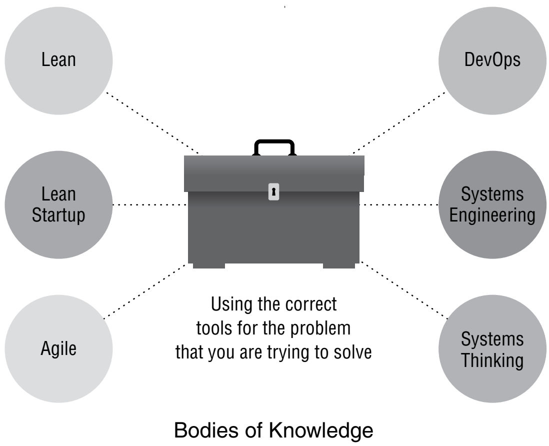


Figure B.2: Industrial DevOps Bodies of Knowledge

Lean

According to the Lean Institute, Lean is a way of thinking about creating needed value with fewer resources and continuous experimentation.[1] Lean concepts date back to Venice in the 1450s, when a process was developed to sequence and standardize the process of galley shipbuilding for shipbuilders.[2] The Venetian Arsenal was said to be able to move ships through their entire production line in an hour.

In 1910, Henry Ford developed his manufacturing strategy for the automobile, where he arranged people, machines, equipment, tools, and products to obtain a continuous flow of production. While Ford was not the inventor of the automobile, what he did was invent an approach to improve the flow of work with the moving assembly line and the five-dollar workday.[3] The assembly line, the use of the conveyor belt, and streamlined practices improved the production of the automobile, making it more affordable for more people. Not only did his strategy result in more affordable cars, but the return to the company resulted in increased wages and improved living for the employees.[4]

The concept of Lean evolved with Taiichi Ohno in the 1950s based on the success patterns and principles of the well-known Toyota Production System (TPS). It was further enhanced by W. Edwards Deming's Total Quality Management System. In the 1990s, James P. Womack released *The Machine That Changed the World*, based on his extensive studies of the Toyota Production System in Japan.[5] Womack showed that the Toyota Production System was applicable to any company in any industry in any country. His team searched for a term that would describe its universal nature and the name "Lean" stuck.

Lean, illustrated in Figure B.3, has five key processes: identify value, map the value stream, create flow, establish pull, and seek perfection. The most important concept in the Lean process is customer value. Key benefits obtained from using the Lean process are increased efficiency, reduced waste, increased productivity, and increased product quality. The goals is to pursue perfection.

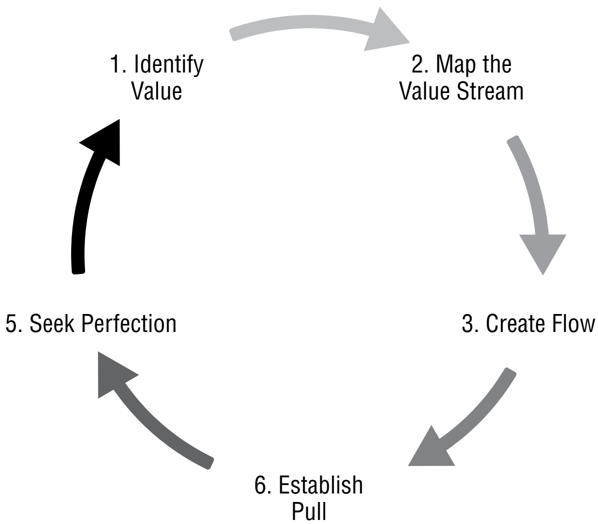


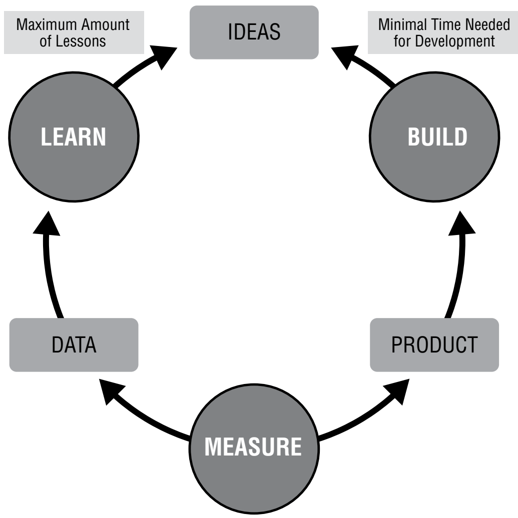
Figure B.3: Lean Production Cycle

Lean Startup

The concept of Lean Startup originated in the early 2000s with Steve Blank and Eric Ries and evolved into a methodology by 2010. Eric Ries went on to publish *The Lean Startup*, a book for entrepreneurs to use continuous innovation and learning to bring innovative products to market rapidly. Eric describes a build/measure/learn cycle, illustrated in Figure B.4, which is simply the scientific method reimagined for business.

Instead of one hypothesis, as we see in the scientific method, Ries promotes two hypotheses, which are a value hypothesis (What problem are we trying to solve?) and a growth hypothesis (How can I scale the benefit?). Next, we run an experiment with the fastest, cheapest way to validate the hypothesis. During the experiment, he recommends not asking for opinions but rather observing people's behavior about how they interact with your product. Based upon observations, adjust the product, with the goal being to complete the build-measure-learn loop as fast as possible. (The process he describes very much aligns with how we train algorithms in machine learning.)

Figure B.4: Build, Measure, Learn



Another key concept that Ries defines is a minimum viable product (MVP), which he defines as a product with just enough features to learn from.[6] This concept is often misunderstood across many domains, where people interpret MVP as something releasable to the customer. For example, instead of a cardboard prototype or model that allows quick validation of our ability to fit various smartphones into a car compartment, we wait until we can actually release a physical product into the vehicle. This approach to the MVP delays our learning and reduces our ability to adapt.

Value Stream Management

The concepts of value stream management were published in the book *Project to Product* by Mik Kersten in 2018. *Forbes* defines value stream management as "a lean business practice that helps determine the value of software development and delivery efforts and resources."[7]

Value stream management leverages techniques such as value stream mapping that were popularized in the 1980s by James Womack and Dan Jones in regard to their work on the Toyota Production System in the 1980s.

Agile

Agile is not one single principle or practice but a product development life cycle, as waterfall is a product development life cycle:

- **Waterfall**: predictive, synchronous, phase-gate delivery mechanism
- **Agile**: empirical, iterative, incremental delivery mechanism

Agile is an evolution of an iterative and incremental approach to managing work that was first described in the 1930s, when the physicist and statistician Walter Shewhart of Bell Labs applied what he referred to as Plan-Do-Study-Act (PDSA) cycles to the improvement of products and processes.[8] Multiple practitioners, including W. Edwards Deming, further evolved the approach to developing products.

In 1986, Hirotaka Takeuchi and Ikujiro Nonaka published "The New New Product Development Game," where they compared product development to a game of rugby.[9] They discussed the team as a Scrum that operates as a single unit moving the ball down the field to accomplish their goal. They defined development as a holistic approach to building products that outline six characteristics: built-in instability, self-organizing project teams, overlapping development phases, "multi-learning," subtle control, and organizational transfer of learning. You can see the initial instantiation of this development life cycle had nothing to do with software.

As is common in all things, what is old became new once again when a group of software professionals collaborated to build a better approach to software development. This group defined the Agile Manifesto, illustrated in Figure B.5, to minimize challenges associated with software development.

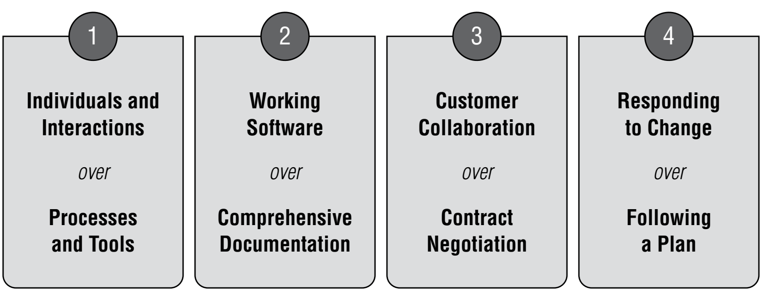


Figure B.5: Agile Values

The Agile Manifesto was written in 2001 and promoted four core values: (1) individuals and interactions over processes and tools, (2) working software over comprehensive documentation, (3) customer collaboration over contract negotiation, and (4) responding to change over following a plan.[10] In addition to the manifesto, the authors agreed to twelve principles, outlined below in Table B.1, to back up the manifesto.

Contrasted with the waterfall phase-gate life cycle, Agile uses short, iterative cycles with frequent customer involvement to incrementally deliver products, resulting in increased adaptability, shorter schedules, reduced cost, increased transparency, and higher employee morale.

Table B.1: Agile Principles Defined

Twelve Principles		
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
Business people and developers must work together daily throughout the project.	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
Working software is the primary measure of progress.	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	Continuous attention to technical excellence and good design enhances agility.
Simplicity—the art of maximizing the amount of work not done—is essential.	The best architectures, requirements, and designs emerge from self-organizing teams.	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Since 2001, Agile has been known as a software development approach utilized to improve software delivery. Numerous benefits have been reported, including increased ability to adapt to change, reduced development schedules, reduced development costs, increased product quality, increased stakeholder transparency, and increased employee morale. Some of the most common frameworks include Scrum, kanban, Lean, eXtreme programming (XP), feature-driven development (FDD), dynamic systems development method (DSDM), and eXtreme manufacturing. Each framework provides a team structure, communication mechanisms, tools, and artifacts. The common characteristics of the frameworks are iterative, incremental, modular, time bound, simple, adaptive, transparent, collaborative, value focused, continuous feedback, and rapid learning.

While there are many common elements within the frameworks, there are many underlying practices, as illustrated below by the Agile Alliance's subway map.[11] Practitioners select the practices that support their solution needs. Many of the practices are interdependent on others. One practice enhances the effects of another. However, it is not recommended to implement them all at once or in their entirety.

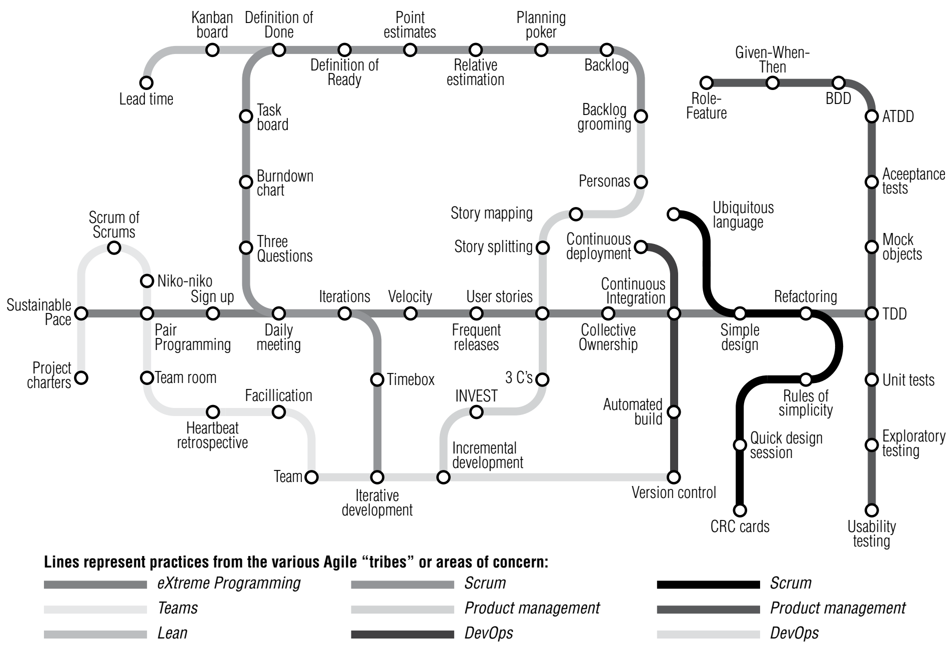


Figure B.6: The Agile Subway

Source: Carignan, Louis-Philippe, "Agile, Is It Just a Delivery Mechanism?"

The initial literature around Agile software development focused on small, cross-functional software teams that were colocated. The success that small Agile software teams achieved led to the question of whether Agile frameworks could be successfully scaled to support many interdependent teams. According to Digital.ai's *16th Annual State of Agile Report*, the Scaled Agile Framework (SAFe) is the most utilized scaling framework, with 53% of the scaling users reporting the use of SAFe. The second-most popular approach is Scrum@Scale, which increased in popularity in 2022 after years of decline. (The full list of popular frameworks is in Table B.2.)

Table B.2: Most Popular Agile Scaling Frameworks

Framework	Author	Date	Used
Scrum@Scale	Jeff Sutherland; Ken Schwaber	1996	9%
Large-Scale Scrum (LeSS)	Craig Larman; Bas Vodde	2005	3%
Agile Portfolio Management	Jochen Krebs	2008	3%
Scaled Agile Framework (SAFe)	Dean Leffingwell	2011	37%
Disciplined Agile (DA)	Scott Ambler	2012	3%
Spotify	Henrik Kniberg; Anders Ivarsson	2012	5%
Enterprise Scrum	Mike Beedle	2013	6%
Unknown/Other	N/A	N/A	23%

Similar to the team-level frameworks, each one of the frameworks provides organizational structure, communication mechanisms, tools, and artifacts. The common characteristics of the frameworks at scale—that is, requiring the coordination of many agile teams across multiple functional areas to deliver an integrated product—in addition to the team-level ones are team coordination, organization hierarchy, system architecture, dependency management, requirements management, value stream management, and the integration of non development functions.

Multiple papers and studies have compared and contrasted the scaling frameworks. Based on that analysis, we have found that at the practice level, there is a significant amount of overlap and agreement in the recommended practices. Therefore, the Industrial DevOps principles defined in this book align with any of the scaled frameworks.

DevOps

DevOps evolved from a series of events and was built upon movements that had come before, including Lean and Agile.

In 2009, Patrick Debois popularized the term *DevOps* at a Velocity event in Belgium. Development and Operations teams had a dysfunctional relationship, which resulted in large delays and defects. Agile software development exacerbated this existing problem. The two domains had never been aligned; Development teams are incentivized to deliver frequently, which maximizes change, and Operations are incentivized to keep the operation baseline stable, which minimizes change. Once software started being deployed faster, the relationship became untenable.

The solution itself was a simple one. Align Development and Operations with a common set of incentives, which was to deliver fast while keeping the operational baseline stable. The result of this simple change was shorter lead times, lower costs, and increased levels of quality. Now, over a decade later, there have been countless books to describe this cooperation between Development and Operations to deliver capability rapidly to the user, with the most impactful being *The Phoenix Project* in 2013.

For purposes of this discussion, we define DevOps as a "mixture of people, process, and technologies that provides a delivery pipeline enabling organizations to move both responsively and efficiently from concept to business outcome." This aligns with the IEEE standard definition of DevOps as "a set of principles and practices emphasizing collaboration and communication between software development teams and IT operations staff along with acquirers, suppliers, and other stakeholders in the life cycle of a software system."[12] The principles of DevOps permeate across the DevOps software pipeline, starting with the team, through release into production.

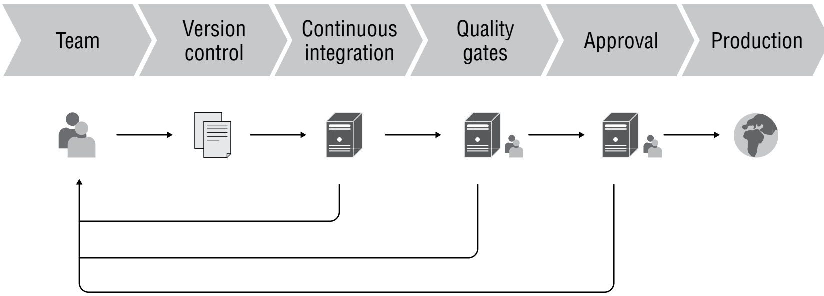


Figure B.7: DevOps Pipeline

Systems Thinking

Systems thinking is a holistic approach to viewing a system through its constituent parts as well as how those parts interrelate with one another and within the context of larger systems. The concept of systems thinking emerged in 1956, when Professor Jay Forrester founded the System Dynamics Group at MIT's Sloan School of Management. Through his experience building aircraft simulators and building computerized combat systems, he learned that the biggest impediments to problems were not on the engineering side but on the management side. He believed that this was because social systems are far more complex than physical systems. Through the System Dynamics Group, Forrester was able to mathematically model complex issues and problems.

Figure B.8: Causal Loop

Systems thinking provides us with a variety of tools and methods, such as the causal loop illustrated in Figure B.8, which allows us to model cause and effect.

Another key tool of systems thinking is the iceberg model, which begins with the observation of events or data to identify patterns over time. This approach, outlined in Figure B.9, surfaces the underlying structures that drive the events and patterns.

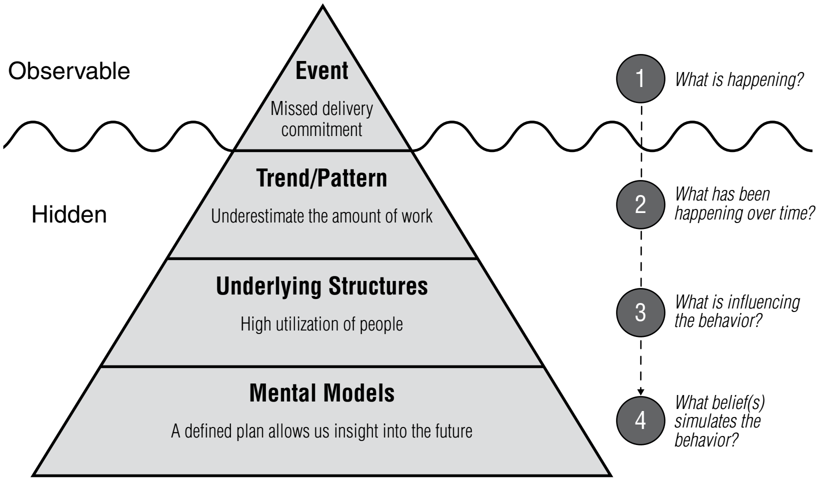


Figure B.9: Systems Thinking Iceberg

Systems thinking is a critical tool to help people solve complex problems through observation and feedback loops, allowing us to view the system as a whole instead of just the parts.

Systems Architecture

There is not a common understanding of the definition of architecture across different domains and industries, so before we tell you why architecture is important to systems, we must begin by developing a common understanding of terms such as *architecture* and *systems*.

The fifth edition of the *Shorter Oxford English Dictionary* begins by defining *architecture* as "the art or practice of designing and constructing buildings," which is not surprising, given the origins. Architecture dates back to approximately 10,000 BC, when humans moved out of caves and began building physical structures to meet a set of needs, such as shelter from the weather in areas where food was plentiful. Later, these needs became more complex. When we began growing our food, these structures needed to be built to leverage the local elements to provide safety and comfort and to store food.

Choice education further abstracted to "the complex or carefully designed structure of something."

In our case, we are designing and constructing *systems*. So, what's a system?

Going back to our trusty dictionary, a *system* is "a set of things working together as parts of a mechanism or an interconnecting network." Some examples of systems include your phone and your vehicle, but they also include things you may not have considered, such as the human brain or society as a whole. Systems can be categorized as *natural systems* or *designed systems*. For this book, we are focused on designed systems.

Given our definition of *architecture* as the complex designed structure of something and our definition of *system* as a set of things working together, we can assume that to effectively meet our needs, we need to *intentionally* design how a set of things work together. This is the basic definition of *systems architecture*.

While systems architecture may sound simple, most organizations do not invest enough in architecting their systems for speed, which includes people, processes, and tools. All systems have an architecture, but if it is not *intentional*, we will not have the culture, process, or technical architecture to meet our unique needs. The layers of the system that need to be intentionally architected include data, application, technical, culture, system, performance, and security, which are illustrated in Figure B.10.

There is agreement that organizations that deliver capabilities faster learn faster, giving them a strong competitive advantage in the market. But there is not enough discussion about how to intentionally architect our systems to enable this speed. That is what we're looking at in this appendix.

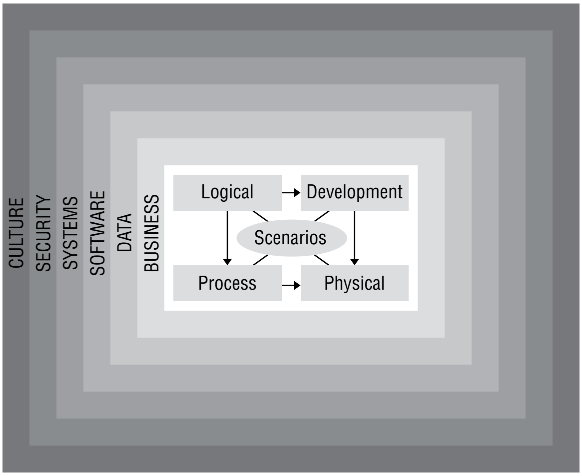


Figure B.10: Intentionally Architected Views

Systems Engineering

The International Council on Systems Engineering (INCOSE) is a not-for-profit organization founded in 1990 that works to advance the state of systems engineering (the practice of architecting designed systems). INCOSE defines engineered (designed) systems as a system designed or adapted to interact with an anticipated operational environment to achieve one or more intended purposes while complying with applicable constraints.[13]

A *basic system*, illustrated in Figure B.11, exists in an environment and has a boundary, inputs, processes, and outputs. For example, a thermometer is a basic engineered system. If a thermometer exists in Italy (*environment*) and senses the local temperature (*boundary*), that temperature is *input* through the sensor and the thermometer calculates a number (*process*) and *outputs* degrees Celsius.

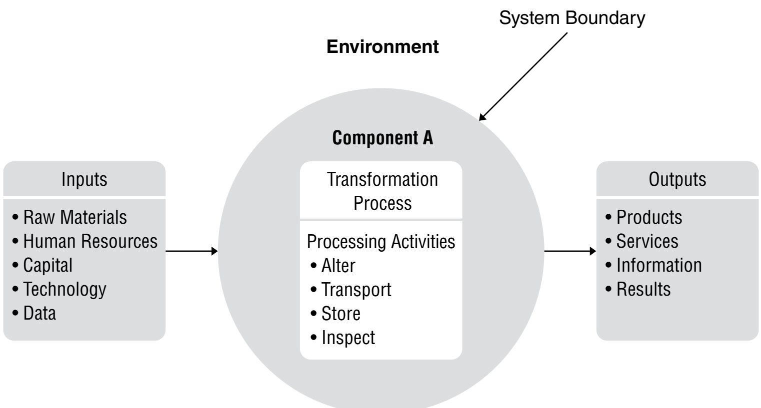


Figure B.11: Basic System

There are three basic types of systems: open, closed, and isolated. An open system can exchange both energy and matter with its environment, a closed system can exchange energy but not matter, and an isolated system is where neither energy nor matter can be exchanged. A thermometer is considered a basic *closed* system.

A *complex system* has multiple components and additional feedback loops, illustrated in Figure B.12, and requires additional work to make updates. Consider a thermometer that senses both temperature *and* humidity. It interacts with two or more components in the system and outputs readings in degrees for temperature as well as the percent of humidity in the air.

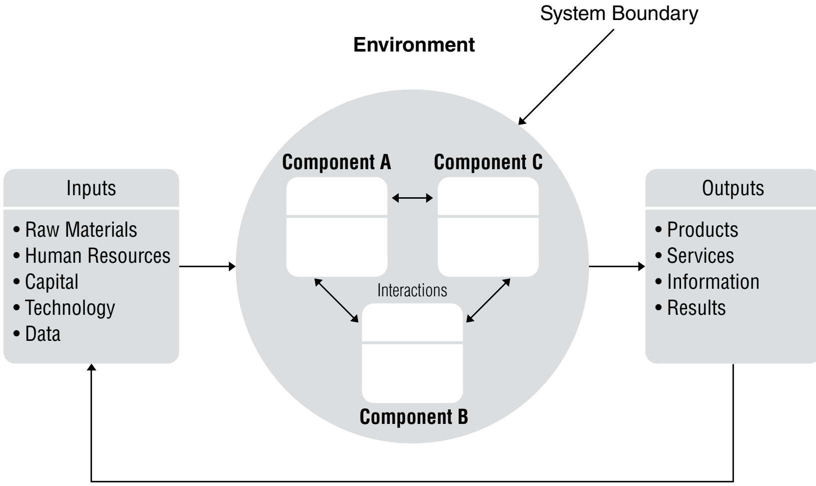


Figure B.12: Closed System with Feedback

The next level of complexity is a *system of systems*, which is a collection of systems that pool resources together to create a more complex system with increased capabilities (illustrated in Figure B.13). Let's consider the average digital thermometer that controls the heating and cooling of your home. These thermometers often have additional components that not only read the temperature but compare the temperature to low and high preset threshold settings. The thermometer not only outputs readings in degrees but also determines whether to trigger heating or air-conditioning systems based on the thresholds that have been set.

If the common thermometer that controls the heating and cooling of your house is a *complex system of systems*, how would you describe a car? A satellite? A collection of satellites? Or SpaceX's Starlink satellite internet constellation? Given the level of complexity of these systems and their criticality to our daily lives, an intentionally disciplined approach to architecture across all of the system layers is required to meet the needs of all system stakeholders and especially our customers.

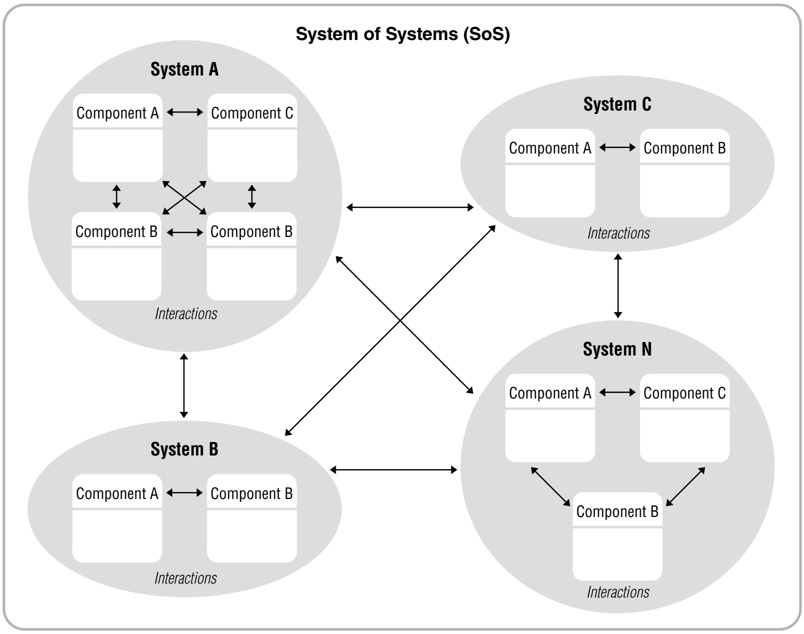


Figure B.13: System of Systems

What's an Interface?

Systems are connected through a series of system *interfaces*, which establish physical connections between systems with a common messaging syntax so that data is understood by both systems. The fundamental aspect of an interface is functional and is defined as the inputs and outputs of functions. Physical interfaces are functions that are performed by physical elements (system elements), and inputs/outputs of functions are also carried by physical elements.

Both functional and physical aspects are considered when designing interfaces. A detailed analysis of an interface shows the function "send" located in one system element, the function "receive" located in the other one, and the function "carry" as being performed by the physical interface that supports the input/output flow.

The architectural interface embraces the concept, allowing its design to be conceived as elements that run independently from each other to allow technological progress without compromising the system.

Architecture of interfaces requires extensive analysis of a variety of trade-offs including performance, scalability, reliability, availability, extensibility, maintainability, manageability, and security. There is no one best practice; it's key to understand your system of interest (SOI) and the context in which it will exist.

Conceptual Models

A systems engineer will start the architecture process with a series of conceptual models that help the engineer to reason out the types of structures and behaviors that are necessary for the system they are designing. The models include (in order) logical architecture, process architecture, development architecture, physical architecture, and scenarios and use cases:

Logical architecture: This is a representation of the structure of the system technology concepts. The goal is to be able to communicate the architecture of the system and understand the intent without having to make specific technology choices. Logical architecture can be used to describe the functionality of the system to the end user. The two most common diagrams utilized for logical architecture are class diagrams (which represents the structure of the system) and state diagrams (which represents the behavior of the system).

Process architecture: After the logical architecture has been established, systems engineers produce a process definition. Process diagrams communicate dynamic aspects of the system. Common diagrams utilized in process architecture are sequence diagrams (process flow in a time sequence), communication diagrams (interactions between objects), and activity diagrams (workflow with steps and actions).

Development architecture: The development views can be used to share how engineers plan to implement the functionality of the system. The most common diagrams for development are component diagrams (wiring of the software components) and package diagrams (demonstrates dependencies). Package diagrams are an excellent way to communicate dependencies and share the hierarchy of elements.

Physical architecture: This architecture communicates the physical aspect of the systems, such as the connections between components. Deployment diagrams are commonly used to demonstrate where capabilities are located physically and the execution of the architecture of the system.

Scenarios and use cases: Lastly, scenarios and use cases demonstrate the architecture in action. Use cases communicate a flow of activities and typically have multiple personas to demonstrate multiple paths through the system. The paths through the system can be aligned with the backlog, with each path representing one or more features.

Business Architecture

According to Red Hat, "Business architecture is a foundational practice that bridges the gaps between business and technology."[14] Business architecture originated in the 1980s in cross-organizational design but has evolved to become a first-class citizen in the enterprise architecture frameworks.

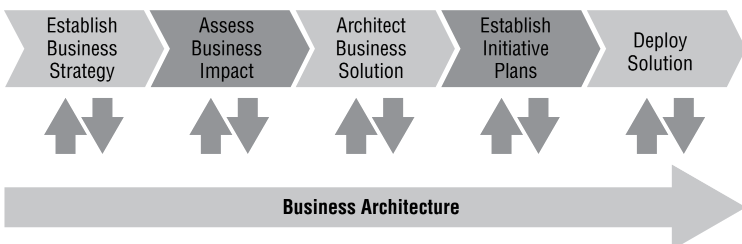


Figure B.14: Business Architecture

Business architecture provides a line of sight from business objectives to execution. This transparency enables alignment across the organization, which is important to be able to deliver value to our stakeholders. How many times have you seen different portions of your organization working toward conflicting goals? The lack of transparency creates excessive waste, hindering delivery.

Data Architecture

While there is no single definition of data architecture, for our purposes, we will define *data* as facts and statistics collected together for reference or analysis and *architecture* as the complex designed structure of something. Therefore, data architecture is the organization of facts and statistics to be utilized for reference or analysis throughout the life cycle.

Over the years, data has moved from siloed and on-premises solutions to open, cloud-based offerings based on the needs of the business, the technology available, and the constraints of the system, as illustrated in Figure B.15. Key considerations in developing the best data architecture include what your budget is, where the sources of your data are, what types of data you have (structured, semistructured, unstructured), what regulations you have on where data is located, how your data will be used (reporting, streaming analytics), and how you need to present the data.

Data architecture exists on a spectrum where one side of the spectrum for data architecture is *on-premises*, *siloed*, and *consolidated data within silos* and the other end of the spectrum is *cloud-based*, *open*, and *federated*. Given society's need for more knowledge faster, it's likely your business needs to continuously move to cloud-based, open, and federated for optimal performance and speed.

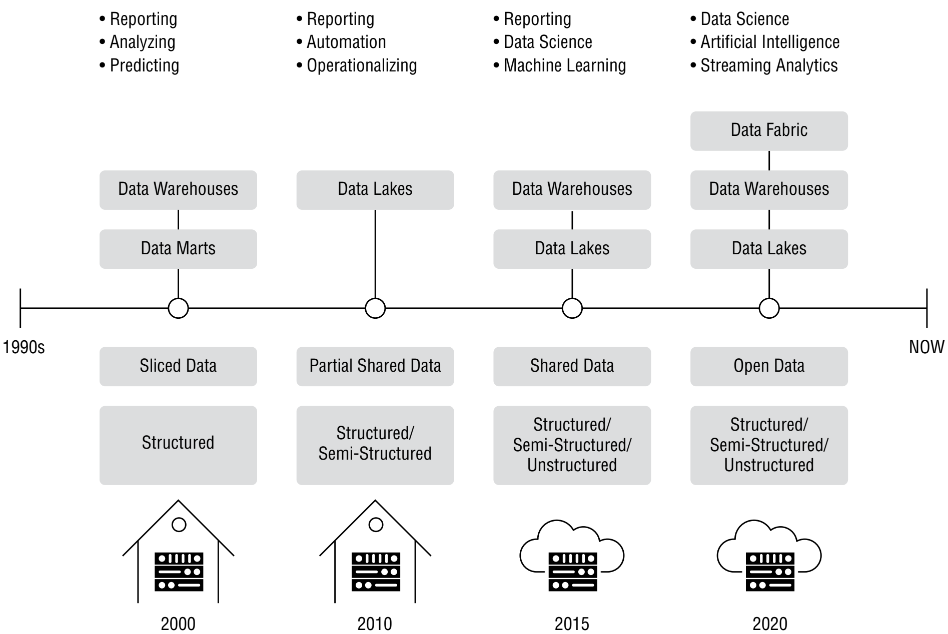


Figure B.15: Data Architecture Evolution

Software/Application Architecture

For our discussion, we are going to use TechTarget's definition of *application architecture*, which they define as a structural map of how an organization's software applications are assembled and how those applications interact with each other to meet business or user requirements.[15] The application architecture evolution illustrated in Figure B.16 has been evolving since the 1980s. Application architectures have evolved from monolithic architecture, which is a single unified unit; to service-oriented architecture, which decomposes the single unit into a set of modules; to microservices, which further decomposes into decoupled units of capability; and finally to serverless, which runs those discrete microservices on demand. Just as with the other architecture domains, the trend is toward modular and loosely coupled capability.

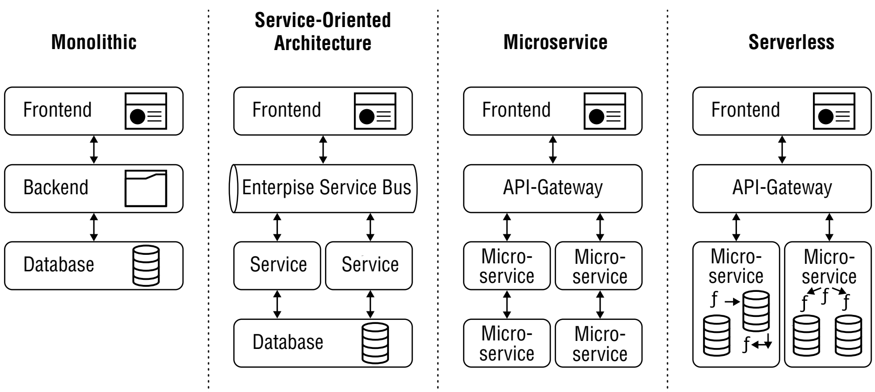


Figure B.16: Evolution of Software Architecture

Software architecture intentionally defines how things connect to ensure we have a scalable, reliable, and available solution. You need to invest in your architecture if the knowledge to build the system exists only in the *wetware* of your teams. Patterns to consider when architecting your application architecture include layered, client-server, event-driven, microkernel application, and microservices:

Layered architecture: Software components are separated into layers of work. The most common is N-tier architecture, which typically has four layers (presentation, business, application, and data). This is a great option if you are building web applications.

Client-server architecture: This pattern has a server and multiple entities. The clients make requests, and the server responds. This pattern is ideal for banking applications.

Event-driven architecture: This pattern involves services that are triggered by an event, such as user interaction. This is an excellent pattern for websites.

Microkernel application architecture: This pattern involves a core application with plug-in modules. This pattern is best for product-based or scheduling applications.

Microservices architecture: This pattern basically builds small services and aggregates them like building blocks to create larger solutions. This pattern can be used on a range of websites to real-time embedded systems.

Design Thinking

Similar to systems thinking, design-thinking concepts were formulated in the 1950s and evolved in the 1960s with what was referred to as *design science*. Design thinking is a human-centered approach to developing products and services. The second wave of design thinking came with Nigel Cross, a human-computer interaction researcher, who published "Designerly Ways of Knowing." The 1990s continued to evolve design thinking concepts with the publishing of "Wicked Problems in Design Thinking."[16]

There are six steps in the design thinking process: *empathize*, *define*, *ideate*, *prototype*, *test*, and *implement* (illustrated in Figure B.17). The steps allow designers to immerse themselves in their customers' experiences to provide better products.

We have all experienced products that were not intuitive and created problems instead of solving them. The body of knowledge built around design thinking allows everyone to be creative designers, instead of a rare few, with a few proven principles defined next:

1. **Empathize:** Understand the problem from your customer's viewpoint through observation and engagement.
2. **Define the problem:** Create a clear statement of the problem we are trying to solve.
3. **Ideate:** Brainstorm solution ideas.
4. **Prototype:** Build a tactile, low-cost solution to the problem.
5. **Test:** Get feedback from your prototyped solution.
6. **Implement:** Build the final solution.

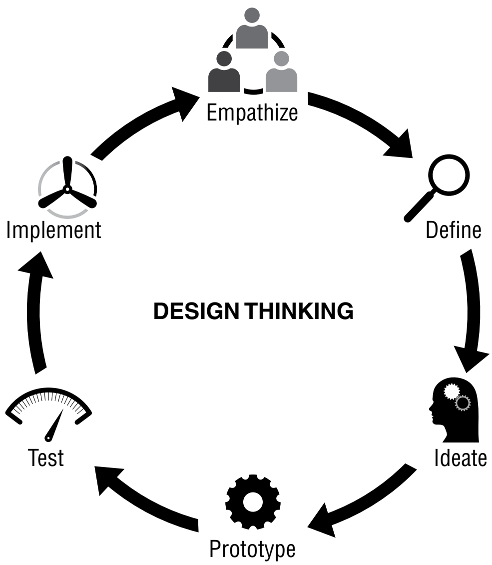


Figure B.17: Six Steps of Design Thinking

Harvard Business Review describes how people are rooted in status quo or behavioral norms that block our ability to imagine new possibilities.[17] Design thinking has allowed many businesses to overcome these challenges to build superior solutions.

Digital Engineering

According to the Software Engineering Institute, digital engineering is defined as an "integrated digital approach that uses authoritative sources of systems data and models as a continuum across disciplines to support life cycle activities from concept through disposal."[18] Digital engineering encompasses a number of methods and tools, including modeling, digital twins, augmented reality, virtual reality, and quantum computing, to name just a few. Digital engineering has exploded across various domains, from aerospace and automotive to energy and health care. The common denominator is that they have to reduce the cost and schedule of products while maintaining safety and security.

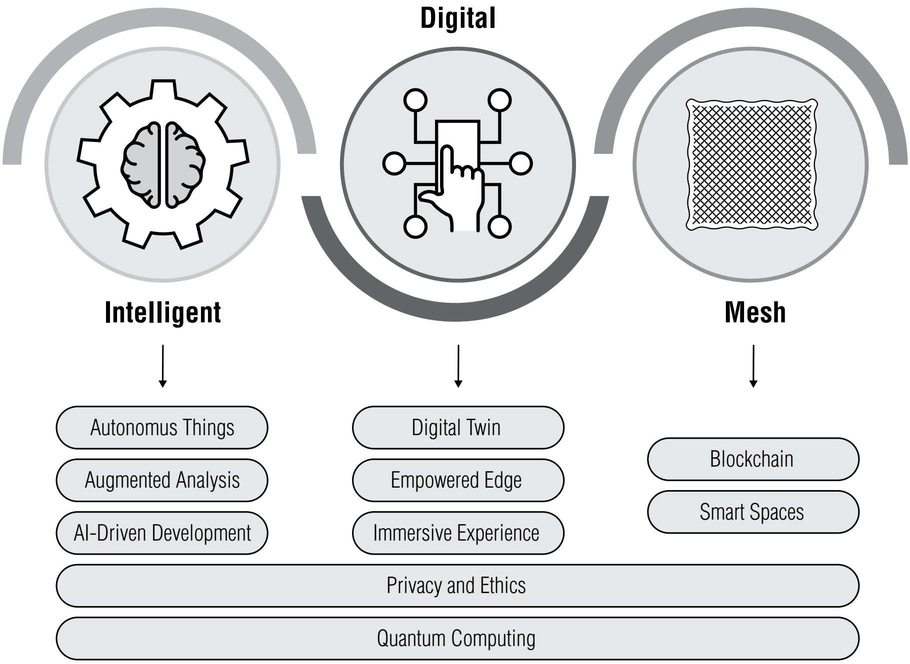


Figure B.18: Digital Engineering

The US Department of Defense (DoD) builds some of the most complex bespoke systems in the world. To that end, they have adopted a digital engineering approach to modernizing how the DoD designs, develops, delivers, operates, and sustains systems. In 2019, Philomena Zimmerman from the Office of the Under Secretary of Defense for Research and Engineering wrote extensively about the DOD's Digital Engineering Strategy and outlined its goals, outlined in Figure B.19. The goals of digital engineering are to provide increased visibility, accuracy, and adaptability for physical systems and their stakeholders.

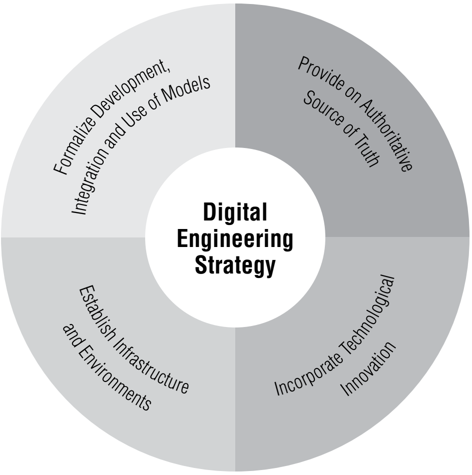


Figure B.19: DoD Digital Engineering Goals

Agile Manufacturing

Lean principles have been well grounded in manufacturing for decades. Lean focuses on reducing waste and improving flow and sustaining it. Sustaining products to keep their value requires a continuous improvement mindset, another core component of Lean. Henry Ford was also interested in how to improve flow and productivity. This led him to implement and use the conveyor belt on the assembly line, which resulted in significant gains in manufacturing production. The Lean community is entrenched in improving flow with additional practices that enable flexibility and adaptability.

The combination of Lean with Agile principles has shifted parts of manufacturing to focus not only on flow and the delivery of products but on how to build better systems that are faster, more flexible, and more adaptive. This is known as Agile manufacturing and eXtreme manufacturing, which are similar in principle.

eXtreme manufacturing started in 2006 with Joe Justice of Wikispeed. Wikispeed designed and manufactured cars for the road and the racetrack. Wikispeed set records for speed of development and won on of the most famous car races in the world, the Nürburgring 24 Hours in Germany.

The Agile practices used included modular architecture, mob development, and eXtreme programming from the software world. These practices have evolved into a set of principles and practices for hardware and in a domain area with high safety requirements and a highly regulated environment. This approach to manufacturing products creates an environment with digital tools and designs, providing the ability to adapt to new priorities and technologies quickly. Justice has since gone on to work at Tesla and shares Agile hardware practices globally with books, blogs, keynotes, conferences, classes, interviews, and more.

In the publication *Agile Manufacturing*, Nicola Accialini defines this concept as "being able to offer a greater production mix using fewer resources."[19] Agile manufacturing includes faster product development cycles and the ability to develop the factory as an Agile manufacturing system. The smart factory is a cyber-physical system. As a result, we are using cyber-physical systems (aka the factory) to build cyber-physical systems (e.g., cars, spacecraft).

The Agile manufacturing system is reconfigurable quickly "according to the production mix and market demands, with high-level autonomy."[20] The ability to achieve mass customization in the factory is built on the foundation and mindset of innovation, training, collaboration, and leadership enabled through three pillars: Modularity and Flexibility, Lean and Six Sigma, and Automation/Industry 5.0, as depicted by Accialini in Figure B.20.

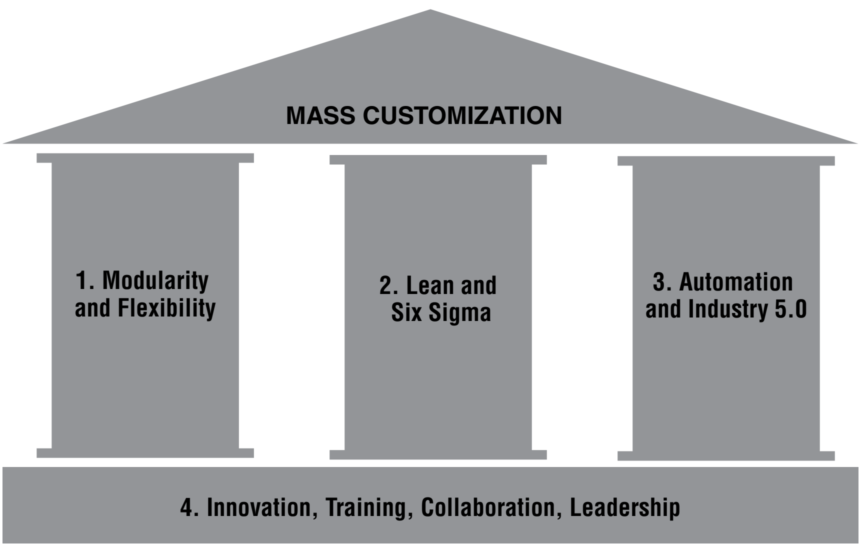


Figure B.20: The Three Pillars of Agile Manufacturing Systems

Source: Nicola Accialini, Agile Manufacturing*.*

Used with permission.

Additive Manufacturing

Additive manufacturing (AM) is defined by ASTM International as "a process of joining materials to make objects from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing methodologies."[21] Additive manufacturing began in the late 1980s with stereolithography (SLA) from 3D Systems, a process that solidifies thin layers of ultraviolet (UV) light-sensitive liquid polymer using a laser. This process of incrementally building up a three-dimensional object in layers is also referred to as 3D printing. The 3D printing process is illustrated below in Figure B.21.

The 3D printing steps are:

1. Create a 3D model.
2. Convert the 3D model into an .STL file that can be sliced into thin layers.
3. Transfer the .STL file to the 3D printer.
4. Set up the machine (3D printer) with configuration parameters and materials.
5. Build the product per motion coding.
6. Remove the product from the printer.
7. Complete post-processing tasks (cleaning, polishing, painting).

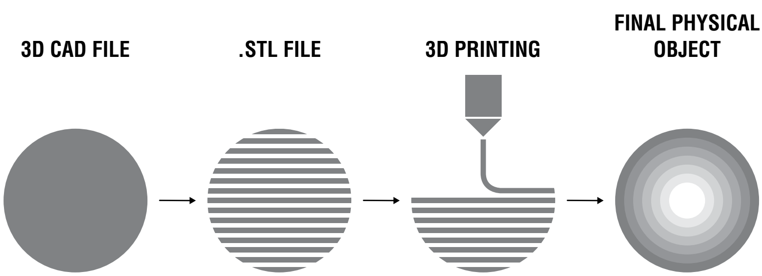


Figure B.21: 3D Printing Steps

Additive manufacturing differs from traditional manufacturing in incrementally adding materials to build a product. In contrast, traditional manufacturing removes materials to build a product. While additive manufacturing has many benefits, the speed of prototype development and the speed of prototype for production allow a much faster feedback loop for physical systems.

APPENDIX C

TOOLS: QUICK REFERENCE GUIDE

Throughout this book, many tools and techniques have been shared that can help you along your journey. While our list is not all inclusive, it provides a foundational set that has been implemented in many organizations. Table C.1 summarizes the Industrial DevOps principles, several of the relevant tools or techniques, and when to use them.

Table C.1: Quick Reference Guide

Principle	Tool or Technique	When to Use
Principle 1: Organize for the Flow of Value	Organizational structure analysis	Pick the best structure for your organizational goals.
	Value stream mapping	Define the mapping of flow from customer need to product delivery. To find bottlenecks in flow.
	Team Topologies composition	Defining the team structures to consider different team types.
Principle 2: Apply Multiple Horizons of Planning	Lean canvas	Capture business needs, a solution summary, and benefits.
	Road mapping	Define high-level goals over time and connect strategy to execution.
	Patterns of decomposition	Decompose the system functionality to fit into timeboxes.

Principle	Tool or Technique	When to Use
Principle 3: Implement Data-Driven Decisions	Objective and Key Results	Set high-level strategic objectives for the organization with defined targeted results.
	Flow Metrics	Identify, measure, and analyze the workflow through your system.
	Variety of digital tools such as 3D printing, prototypes, digital and simulated models, digital shadows, digital twins, and emulators.	Decide on the tools needed to shift testing and manufacturing left. May be part of investment planning as tools evolve.
Principle 4: Architect for Change and Speed	Modular Open Systems Approach (MOSA)	Enhance interoperability, flexibility, scalability, and affordability by promoting modularity and standardization .
	Artificial Intelligence and Machine Learning	Allow architects and systems engineers to evaluate multiple design options with multiple parameters and remove bad options quickly while amplifying engineering knowledge. Improve predictive analytics and forecasting.
	Digital Twin	Reduce risk by testing in a virtual environment before deploying new capabilities. Improve predictive maintenance.
Principle 5: Iterate, Manage Queues, Create Flow	Flow Charts and Visualization Tools	Visualize the flow of the system and find bottlenecks
	Experimentation toward a Solution	Isolate bottlenecks. GAIL iterative learning and feedback. Test alternate design decisions.
	Set-Based Design	Explore multiple sets of possibilities at the subsystem level against broad targets and proactively explore the limits of hardware design.

Principle	Tool or Technique	When to Use
	Kanban	Visualize and manage queues and the flow of work through the system.
Principles 6: Establish Cadence and Synchronization for Flow	Cadence Analysis (consider factors such as team size, availability, complexity of work, and the need for coordination and feedback)	Determine team and program cadence for planning and demonstrations.
	Program Calendar of Events	Schedule recurring planning and demonstration events for the next six months to a year out.
Principle 7: Integrate Early and Often	CI/CD Pipelines	Automate and streamline product development process.
	Incremental Integration	Detect integration issues and provides opportunities for frequent testing and feedback.
Principle 8: Shift Left	Build Labs Early and Evolve (e.g., software-in-the-loop, hardware-in-the-loop, digital twins)	Enable early and incremental development and testing with improved quality.
	Behavior-Driven Develop-ment	Ensure you are building the right thing and that you are building the thing right.
	Shift-Left Manufacturing	Enable regular feedback loops to improve verification in hardware and manufacturing design and reduce rework further downstream. Improve quality, which reduces cost of rework.
	Testing Strategy	Define your testing approach and invest in the digital environments required to meet quality standards and data needs.

Principle	Tool or Technique	When to Use
Principle 9: Apply a Growth Mindset	Cultural Surveys	Determine current cultural beliefs.
	Intentionally Architect Culture	Drive the organization to implement new behaviors
	Recognition Program	Provide a program where peers can recognize and elevate each other's successes.
	Psychological Safety	Create an environment where people feel free to speak up, share their ideas, share risks, and share failures, which results in increased innovations, learning, and successes.
	T-Shaped Skills	Help teams to deliver faster and fill gaps. T-shaped skills and cross-domain learning are critical.
	Learning Strategy	Shape a common language and shared mental models across leaders, functions, and teams as the new practices become part of the organization's culture.
	Coaching	Help the team and organization improve their practices to deliver value with measurable results while building high-performing teams. This can be Lean coaches, Agile coaches, or leaders/managers as coaches.

Bibliography

"2022 Global Space Industry Report." *Benchmark International* (blog). September 23, 2022. https://blog.benchmarkcorporate.com/2022-global -space-industry-report.

Accialini, Nicola. *Agile Manufacturing: Strategies for Adaptive, Resilient and Sustainable Manufacturing*. Self-published, December 2, 2022.

Afifi-Sabet, Keumars. "How BMW Embraced Agile to Hit New Speeds." IT Pro. August 9, 2018. https://www.itpro.com/agile-development/31552/how-bmw -embraced-agile-to-hit-new-speeds.

Agile Alliance. "Subway Map to Agile Practices." AgileAlliance.org (website). Accessed May 21, 2023. https://www.agilealliance.org/agile101/subway-map-to-agile -practices/.

Agile Manifesto. "Manifesto for Agile Software Development." AgileManifesto.org (website). Accessed May 1, 2023. https://agilemanifesto.org/.

———. "Principles behind the Agile Manifesto." AgileManifesto.org (website). Accessed May 1, 2023. https://agilemanifesto.org/principles.html.

Amazon. "Who We Are." AboutAmazon.com (website). Accessed May 7, 2023. https:// www.aboutamazon.com/about-us.

Amazon Web Services. "Formula 1 Redesigns Car for Closer Racing and More Exciting Fan Experience by Using AWS HPC Solutions." AWS.Amazon.com (website). 2021. https://aws.amazon.com/solutions/case-studies/formula-1-graviton2/.

Anderson, Katie. *Learning to Lead, Leading to Learn: Lessons from Toyota Leader Isao Yoshino on a Lifetime of Continuous Learning*. California: Integrand Press, 2020.

Anderson, Patrick. "Code on the Road: BMW's Global Value Stream Management (VSM) Journey." *Plainview Blog*. January 27, 2021. https://blog.planview.com /code-on-the-road-bmws-global-value-stream-management-vsm-journey/.

Apple, Inc. "Apple Inc. Form 10-K for the Fiscal Year Ended September 30, 2017." Accessed May 7, 2023. https://s2.q4cdn.com/470004039/files/doc_ financials/2017/10-K_2017_As-Filed.pdf.

Apposite Technologies. "Satellite Testing: Best Practices for Application Performance Testing Over Satellite." Accessed May 14, 2023. https://www.apposite-tech.com/ blog/best-practices-to-test-application-perfor mance-over-satellite-networks/.

Augustine, Sanjiv, Roland Cuellar, and Audrey Scheere. *From PMO to VMO: Managing for Value Delivery*. Oakland, CA: Berrett-Koehler Publishers, 2021.

Automotive World. "Bosch: Did You Know. . . . ." *Automotive World* (website). July 21, 2020. https://www.automotiveworld.com/news-releases/bosch-did-you-know/.

Bellan, Rebecca. “Joby Aviation’s Contract with US Air Force Expands to include Marines.” *TechCrunch*. August 10, 2022. https://techcrunch.com/2022/08/10/joby-aviations-contract-with-u-s-air-force-expands-to-include-marines/.

Berg, Cliff. “SpaceX’s Use of Agile Methods.” Medium. December 9, 2019. https://cliffberg.medium.com/spacexs-use-of-agile-methods-c63042178a33.

Biggs, Renee. “What Is a Business Architect and How Do You Become One?” Red Hat. December 16, 2022. https://www.redhat.com/architect/business-architect-career#:~:text=Business%20architecture%20is%20a%20foundational,gaps%20between%20business%20and%20technology.&text=A%20business%20architect%20is%20a,the%20business%20and%20its%20users.

Blinde, Loren. “Lockheed Martin Completes First LM 400 Multi-MissionSpacecraft.” Intelligence Community News. January 31, 2023. https://intelligencecommunitynews.com/lockheed-martins-completes-first-lm-400-multi-mission-spacecraft/.

Bloomberg Technology. “SpaceX Nails Landing of Reusable Rocket on Land.” June 30, 2021. YouTube video, 1:00. https://www.youtube.com/watch?v=Aq7rDQx9jns.

Bosch. “We Are Bosch.” Accessed May 7, 2023. https://www.wearebosch.com/index.en.html.

Brenton, Flint. “What Is Value Stream Management? A Primer For Enterprise Leadership.” *Forbes*. July 8, 2019. https://www.forbes.com/sites/forbestechcouncil/2019/07/08/what-is-value-stream-management-a-primer-for-enterprise-leadership/?sh=1ebb073e7b67.

Brooks, Shilo. “Why Did the Wright Brothers Succeed When Others Failed?” Scientific American. March 14, 2020. https://blogs.scientificamerican.com/observations/why-did-the-wright-brothers-succeed-when-others-failed/.

Buchanan, Richard. “Wicked Problems in Design Thinking.” *Design Issues* 8, no. 2 (1992): 5–21.

Carlos, Juan. “SpaceX: Enabling Space Exploration through Data and Analytics.” Harvard Business School Digital Innovation and Transformation. March 23, 2021. https://d3.harvard.edu/platform-digit/submission/spacex-enabling-space-exploration-through-data-and-analytics/.

Carignan, Louis-Philippe, “Agile, Is It Just a Delivery Mechanism?” Scrum.org (web site). August 27, 2014. https://www.scrum.org/resources/blog/agile-it-just-delivery-mechanism.

Carr, David F. “Digital Transformation Leaders’ Secret: Dump Traditional Org Charts.” The Enterprisers Project. January 3, 2020. https://enterprisersproject.com/article/2020/1/digital-transformation-leaders-how-organize.

Chevron. “The Chevron Way: Getting Results the Right Way.” Accessed May 7, 2023. https://www.chevron.com/about/the-chevron-way.

Cleland-Huang, Jane, Ankit Agrawal, Michael Vierhauser, and Christoph Mayr-Dorn. “Visualizing Change in Agile Safety-Critical Systems.” *IEEE Software* 38, no. 3 (June 2020): 43–51.

Comella-Dorda, Santiago, Lavkesh Garg, Suman Thareja, and Belkis Vasquez-McCall. “Revisiting Agile Teams after an Abrupt Shift to Remote.” McKinsey & Company. April 28, 2020. https://www.mckinsey.com/capabilities/people-and-organizational-performance/our-insights/revisiting-agile-teams-after-an-abrupt-shift-to-remote.

Crook, Joy, Suzette Johnson, Harry Koehnemann, Jeffrey Shupack, Steven J. Spear, Hasan Yasar, Robin Yeman, Jeff Boleng, Eileen Wrubel, and Mik Kersten. *Applied Industrial DevOps 2.0: A Hero's Journey*. IT Revolution DevOps Enterprise Forum. 2020.

Daily, Jim, and Jeff Peterson. "Predictive Maintenance: How Big Data Analysis Can Improve Maintenance." In *Supply Chain Integration Challenges in Commercial Aerospace*. December 2016. https://link.springer.com/chapter/10.1007/978-3-319-46155-7_18.

Datta, Ganesh. "Best of 2022: How DORA Metrics Can Measure and Improve Performance." DevOps.com. December 30, 2022. https://devops.com/how-dora-metrics-can-measure-and-improve-performance/.

Deming, W. Edwards. *Out of the Crisis*. Cambridge, MA: The MIT Press, 2000.

Deutschman, Alan. "Change or Die." *Fast Company*. May 5, 2005. https://www.fastcompany.com/52717/change-or-die.

Devalekar, Ashish. "Why High Employee Engagement Results in Accelerated Revenue Growth." *Forbes*. July 14, 2021. https://www.forbes.com/sites/forbesbusinesscouncil/2021/07/14/why-high-employee-engagement-results-in-accelerated-revenue-growth/?sh=6b1f5952597b.

Digital.ai. *15th Annual State of Agile Report*. Accessed May 22, 2023. https://digital.ai/resource-center/analyst-reports/15th-state-of-agile-report/.

Digital.ai. *16th Annual State of Agile Report*. Accessed April 30, 2023. https://info.digital.ai/rs/981-LQX-968/images/AR-SA-2022-16th-Annual-State-Of-Agile-Report.pdf.

Digital Engineering 24/7. "Simulation Advances NASCAR Race Car Development." November 5, 2020. https://www.digitalengineering247.com/article/simulation-advances-nascar-race-car-development/.

Dikert, Kim, Maria Paasivaara, and Casper Lassenius. "Challenges and Success Factors for Large-Scale Agile Transformations: A Systematic Literature Review." *Journal of Systems and Software* 119 (2016): 87–108.

Doerr, John. *Measure What Matters: How Google, Bono, and the Gates Foundation Rock the World with OKRs*. New York: Portfolio/Penguin, 2018.

Donahue, Colum. "How to Keep Pace with Digital Transformation and Avoid Becoming Obsolete." *Forbes*. February 12, 2020. https://www.forbes.com/sites/theyec/2020/02/12/how-to-keep-pace-with-digital-transformation-and-avoid-becoming-obsolete/?sh=1ee1f6024861.

Dweck, Carol S. *Mindset: The New Psychology of Success*. New York: Ballantine Books, 2006.

Edmondson, Amy C. "Strategies for Learning from Failure." *Harvard Business Review*. April 2011. https://hbr.org/2011/04/strategies-for-learning-from-failure.

Edwards, Damon, Ben Grinnell, Gary Gruver, Suzette Johnson, Kaimar Karu, Terri Potts, and Rosalind Radcliffe. *Making Matrixed Organizations Successful with DevOps: Tactics for Transformation in a Less Than Optimal Organization*. IT Revolution DevOps Enterprise Forum. 2017.

Errick, Kirsten. "Senate Committee Approves AGILE Procurement Act for IT and Communications Tech." Nextgov. August 4, 2022. https://www.nextgov.com

/it-modernization/2022/08/senate-committee-approves-agile-procurement-act-it-and-communications-tech/375404/.
European Space Agency. "Galileo Clock Anomalies under Investigation." January 19, 2017. https://www.esa.int/Applications/Navigation/Galileo _clock_anomalies _under_investigation.
European Space Agency. "No 33–1996: Ariane 501—Presentation of Inquiry Board Report." July 23, 2996. https://www.esa.int/Newsroom/Press_Releases/Ariane_501 _-_Presentation_of_Inquiry_Board_report.
Feldscher, Jacqueline. "China Could Overtake US in Space without 'Urgent Action,' Warns New Pentagon Report." Defense One. August 24, 2022. https://www .defenseone.com/technology/2022/08/china-could-overtake-us-space-without -urgent-action-report/376261/.
Ferguson, Kevin. "Application Architecture." TechTarget. October 2021. https://www .techtarget.com/searchapparchitecture/definition/application-architecture.
Ferreira, Soraia. "Warming to Her Task: How Soraia Ferreira is Giving Heating Technology a Digital Boost." Bosch. Accessed May 1, 2023. https://www.bosch.com /stories/warming-to-her-task/.
Field, Kyle. "Tesla Has Applied Agile Software Development to Automotive Manufacturing." CleanTechnica. September 1, 2018. https://cleantechnica .com/2018/09/01/tesla-has-applied-agile-software-development-to-automotive -manufacturing/.
"Flow Metrics—Understanding Value Stream Metrics." Flow Framework. November 8, 2022. https://flowframework.org/flow-metrics/.
Forbes Technology Council. "11 Benefits of Behavior-Driven Product Development." *Forbes*. August 9, 2019. https://www.forbes.com/sites/forbestechcouncil /2019/08/09/11-benefits-of-behavior-driven-product-development/?sh =75cd9bef1a3c.
Fowler, Martin, and Jim Highsmith. "The Agile Manifesto." *Software Development*. August 2001. http://www.hristov.com/andrey/fht-stuttgart/The_Agile_Manifesto_ SDMagazine.pdf.
Fritzsch, Jonas, Justus Bogner, Markus Haug, Anna Cristina Franco Da Silva, Carolin Rubner, Matthias Saft, Horst Sauer, and Stefan Wagner. "Adopting Microservices and DevOps in the Cyber-Physical Systems Domain: A Rapid Review and Case Study." *Software Practice and Experience* 53, no. 3 (2023): 790–810.
Ford. "The Moving Assembly Line and the Five-Dollar Workday." Accessed May 21, 2023. https://corporate.ford.com/articles/history/moving-assembly-line .html#:~:text=What%20made%20this%20assembly%20line,built%20step %2Dby%2Dste.
Forsgren, Nicole, Jez Humble, and Gene Kim. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*. Portland, OR: IT Revolution, 2018.
Furuhjelm, Jörgen, Johan Segertoft, Joe Justice, and J. J. Sutherland. "Owning the Sky with Agile: Building a Jet Fighter Faster, Cheaper, Better with Scrum." Scrum Inc. 2017. https://www.scruminc.com/wp-content/uploads/2015/09/Release-version_ Owning-the-Sky-with-Agile.pdf.

Gartner. "Use Test-First Development Processes to Jump-Start Your SDLC." December 18, 2019. https://www.gartner.com/en/documents/3978408.

Gemba Academy. "Hoshin Planning." Accessed May 21, 2023. https://www.gembaacademy.com/resources/gemba-glossary/hoshin-planning.

GlobalCom. "The Cost of Building and Launching a Satellite." Accessed May 21, 2023. https://globalcomsatphone.com/costs/.

Gouré, Dan. "DoD's Move to the Cloud Is Critical to Operate at the 'Speed of Relevance.'" RealClear Defense. June 25, 2018. https://www.realcleardefense.com/articles/2018/06/25/dods_move_to_the_cloud_speed_of_relevance.html.

Grant, Adam. *Think Again: The Power of Knowing What You Don't Know*. New York: Viking, 2021.

Groysberg, Boris, Jeremiah Lee, Jesse Price, and J. Yo-Jud Cheng. "The Leader's Guide to Corporate Culture." *Harvard Business Review*.January–February 2018. https://hbr.org/2018/01/the-leaders-guide-to-corporate-culture.

Gruver, Gary, and Tommy Mouser. *Leading the Transformation: Applying Agile and DevOps Principles at Scale*. Portland, OR: IT Revolution Press, 2015.

Hamdi, Shabnam, Abu Daud Silong, Zoharah Binti Omar, and Roziah Mohd Rasdi. "Impact of T-Shaped Skill and Top Management Support on Innovation Speed: The Moderating Role of Technology Uncertainty." *Cogent Business & Management* 3, no. 1 (March 2016).

Harter, Jim. "Is Quiet Quitting Real?" Gallup. September 6, 2022. https://www.gallup.com/workplace/398306/quiet-quitting-real.aspx.

Harvard Business Review Analytic Services. *Competitive Advantage through DevOps: Improving Speed, Quality, and Efficiency in the Digital World*. January 25, 2019. https://hbr.org/resources/pdfs/comm/google/CompetitiveAdvantageThroughDevOps.pdf.

Heistand, Christopher, Justin Thomas, Nigel Tzeng, Andrew R. Badger, Luis M. Rodriguez, Aaron Dalton, Jesse Pai, Austin Bodzas, and Derik Thompson. "DevOps for Spacecraft Flight Software." *2019 IEEE Aerospace Conference* (March 2019): 1–16.

Hessing, Ted. "History of Lean." Six Sigma Study Guide. Accessed May 21, 2023. https://sixsigmastudyguide.com/history-of-lean/.

Howard, Ben. "What Is Agile Aerospace? Learn Planet's Approach." Planet. October 16, 2019. https://www.planet.com/pulse/what-is-agile -aerospace-learn-planets -approach/.

Howell, Elizabeth. "8 Ways that SpaceX has Transformed Spaceflight." Space.com. March 25, 2022. https://www.space.com/ways-spacex -transformed-spaceflight.

Humble, Jez, and David Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Boston: Addison-Wesley, 2010.

IEEE Standards Association. "IEEE 2675-2021: IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment." April 2016. https://standards.ieee.org/ieee/2675/6830/.

INCOSE. "Systems and SE Definitions." INCOSE.org (website). Accessed July 13, 2023. https://www.incose.org/about-systems-engineering/system-and-se-definition/system-and-se-definitions.

International Center for Clinical Excellence. "The Backwards Bicycle: What It Takes to Unlearn Old and Learn New Habits." July 21, 2021. YouTube video, 3:56. https://www.youtube.com/watch?v=CqwfGUhYBEA.

International Organization for Standardization (ISO). "ISO 26262-1:2011 Road Vehicles—Functional Safety—Part 1: Vocabulary." November 2011. https://www.iso.org/standard/43464.html.

Ismail, Nick. "AI Solutions Required for Fast-Paced Application Development?" *Information Age*. July 5, 2018. https://www.information-age.com/ai-assistant-application-development-10672/#:~:text=Gartner%20predicts%20that%20%E2%80%9Cby%202022,what%E2%80%99s%20called%20low%2Dcode%20development.

ISO Update. "What Is Quality." September 23, 2019. https://isoupdate.com/resources/what-is-quality/.

ISS National Laboratory. "From Sample to Results: In-Space Data Analysis Enables Quicker Data Return." April 5, 2022. https://www.issnationallab.org/hpe-sbc2-inspace-data-analysis-enables-quicker-return/.

Johnson, Suzette, Robin Yeman, Harry Koehnemann, Jeffrey Shupack, Matt Aizcorbe, Adrian Cockcroft, Michael Mckay, and Hasan Yasar. *Building Industrial DevOps Stickiness: Applying Insights*. Portland, OR: IT Revolution Press, 2021.

Johnson, Suzette, Diane Lafortune, Dean Leffingwell, Harry Koehnemann, Stephen Magill, Steve Mayner, Avigail Ofer, Anders Wallgren, Robert Stroud, and Robin Yeman. *Industrial DevOps: Applying DevOps and Continuous Delivery to Significant Cyber-Physical Systems*. IT Revolution DevOps Enterprise Forum. 2018.

Johnson, Suzette, Robin Yeman, Harry Koehnemann, Jeffrey Shupack, Hasan Yasar, Ben Grinnell, Deborah Brey, Steve Farley, and Josh Corman. "Overcoming Barriers to Industrial DevOps: Working with the Hardware-Engineering Community." *The DevOps Enterprise Journal* 4, no. 2 (Fall 2022).

Jurkiewicz, Jakub, Ramanathan Yegyanarayanan, Myles Hopkins, Atulya Krishna Mishra, and Sandeep P R. *Whitepaper: Employee Engagement*. Business Agility Institute. December 5, 2019. https://businessagility.institute/learn/whitepaper-employee-engagement/275.

Justice, Joe. *Scrum Master: The Agile Training Seminar for Business Performance (Agile Business Performance from the Agile Business Institute Book 1)*. Self-published, February 4, 2021. Kindle.

Kalenda, Martin, Petr Hyna, and Bruno Rossi. "Scaling Agile in Large Organizations: Practices, Challenges, and Success Factors." *Journal of Software: Evolution and Process* 30, no. 10 (2018).

Kennedy, Michael. *Product Development for the Lean Enterprise: Why Toyota's System Is Four Times More Productive and How You Can Implement It*. Richmond, VA: The Oaklea Press, 2008.

Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. Portland, OR: IT Revolution Press, 2018.

Khan, Faisal. "Evolution of Mars Rovers in the Past 25 Years." Medium. August 12, 2022. https://medium.com/technicity/evolution-of-mars -rovers-in-the-past-25-years-b89ec4fbc43f.

Kharpal, Arjun. "'Pathetic' Performance Has Left U.S. 'Well Behind' China in 5G Race, Ex-Google CEO Eric Schmidt Says." CNBC. February 17, 2022. https://www.cnbc.com/2022/02/17/us-well-behind-china-in-5g-race-ex-google-ceo-eric-schmidt-says.html.

Kim, Gene, Jez Humble, Patrick Debois, and John Willis. *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. Portland, OR: IT Revolution Press, 2016.

Klender, Joey. "Tesla Makes $9,500 per Car, Eight Times as Much as Toyota." TESLARATI. November 9, 2022. https://www.teslarati.com/tesla-toyota-profit-margin-8-times-q3-2022/#:~:text=This%20is%20where%20the%20financials,report%20states%20this%20is%20unconfirmed.

Kohn, Alfie. "Why Incentive Plans Cannot Work." *Harvard Business Review*. September–October 1993. https://hbr.org/1993/09/why-incentive-plans-cannot-work.

Kolodny, Lora. "Joby Aviation Can't Hit Production Targets on Time, According to Short Sellers' Report." CNBC. September 30, 2022. https://www.cnbc.com/2022/09/30/joby-aviation-cant-hit-production-targets-on-time-short-report.html#:~:text=Joby%27s%20projections,by%20the%20end%20of%202024.

Kontoghiorghes, Constantine, Susan M. Awbre, and Pamela L. Feurig. "Examining the Relationship between Learning Organization Characteristics and Change Adaptation, Innovation, and Organization Performance." *Human Resource Development Quarterly* 16, no. 2. (June 22, 2005). 185–212. https://onlinelibrary.wiley.com/doi/abs/10.1002/hrdq.1133.

Kotter, John. "Leading Change: Why Transformation Efforts Fail." *Harvard Business Review*. May–June 1995. https://hbr.org/1995/05/leading -change-why -transformation-efforts-fail-2.

Kowzan, Mateusz, and Patrycja Pietrzak. "Continuous Integration in Validation of Modern, Complex, Embedded Systems." *2019 IEEE/ACM International Conference on Software and System Processes* (2019): 160–164.

Kumar, S. Annand, and R. V. S. Prasad. "Chapter 2 - Basic Principles of Additive Manufacturing," in Additive Manufacturing, edited by M. Manjaiah, K. Raghavendra, N. Balashanmugam, and J. Paulo Davim. Woodhead Publishing, 2021.

LaBerge, Laura, Clayton O'Toole, Jeremy Schneider, and Kate Smaje. "How COVID-19 Has Pushed Companies over the Technology Tipping Point—And Transformed Business Forever." McKinsey & Company. October 5, 2020. https://www.mckinsey.com/capabilities/strategy-and-corporate-finance/our-insights/how-covid-19-has-pushed-companies-over-the-technology-tipping-point-and-transformed-business-forever.

Landau, Peter. "What Is Lead Time? How to Calculate Lead Time in Different Industries." ProjectManager.com (website). March 21, 2023. https://www.projectmanager.com/blog/lead-time-how-to-calculate.

Lambert, Fred. "First Look at Tesla's New Structural Battery Pack that Will Power Its Future Electric Cars." Electrek. January 19, 2021. https://electrek.co/2021/01/19/tesla-structural-battery-pack-first-picture/.

Lawton, C. A. "The History of Lean – Part 1." C. A. Lawton Co. July 10, 2019. https://calawton.com/lean-history-part-1/.

Lean Enterprise Institute. "Cycle Time." Accessed May 25, 2023. https://www.lean.org/lexicon-terms/cycle-time/.

Lean Enterprise Institute. "Value Streams." Accessed May 5, 2023. https://www.lean.org/lexicon-terms/value-stream/.

Lean Enterprise Institute. "What Is Lean?" Accessed May 21, 2023. https://www.lean.org/explore-lean/what-is-lean/.

Liedtka, Jeanne. "Why Design Thinking Works." *Harvard Business Review*. September–October 2018. https://hbr.org/2018/09/why-design -thinking-works.

Lofgren, Eric. "How Saab Uses Agile Principles to Develop the Gripen Fighter." *Acquisition Talk* (blog). November 17, 2021. https://acquisitiontalk.com/2021/11/how-saab-uses-agile-principles-to-develop-the-gripen-fighter/.

Mahmoud, Magdi S., and Yuanqing Xia. "Chapter 13—Secure Estimation Subject to Cyber Stochastic Attacks." In *Cloud Control Systems: Analysis, Design and Estimatio*n, edited by Stephen Ison and Lucy Budd, 373–404. Cambridge, MA: Academic Press, 2020. https://www.sciencedirect.com/science/article/abs/pii/B9780128187012000214.

Marquet, L. David. *Turn the Ship Around! A True Story of Turning Followers into Leaders*. New York: Penguin, 2012.

Maxwell, John C. *Leadershift: The 11 Essential Changes Every Leader Must Embrace*. Nashville: HarperCollins Leadership, 2019.

McDermott, Rose. "Prospect Theory." Britannica. Accessed May 21, 2023. https://www.britannica.com/topic/prospect-theory/additional-info#history.

Mersino, Anthony. "Why Agile Is Better than Waterfall (Based on Standish Group Chaos Report 2020)." Vitality Chicago. November 1, 2021. https://vitalitychicago.com/blog/agile-projects-are-more-successful-traditional-projects/.

Mochari, Ilan. "Why Half of the S&P 500 Companies Will Be Replaced in the Next Decade." *Inc*. March 23, 2016. https://www.inc.com/ubs/five-lessons-from-companies-that-are-making-a-difference.html.

Moorhead, Patrick. "The Past and Present Are Telling of the Future, So What's Next for Planet?" *Forbes*. November 29, 2021. https://www.forbes.com/sites/patrickmoorhead/2021/11/29/the-past-and-present-are-telling-of-the-future-so-whats-next-for-planet/?sh=35f3376e41e4.

Morris, David. *Storm on the Horizon: Khafji—The Battle That Changed the Course of the Gulf War*. New York: Presidio Press, 2005.

MotorBiscuit. "3 Toyotas with the Lowest 10-Year Maintenance Costs." December 3, 2022. https://www.motorbiscuit.com/3-toyotas-lowest-10-year-maintenance-costs/.

Muruganandham, Shivaprakash. "Cloud Computing: Ratcheting the Satellite Industry Forward." NSR. June 15, 2020. https://www.nsr.com/cloud-computing-ratcheting-the-satellite-industry-forward/.

NASA Science Mars Exploration. "NASA Readies Perseverance Mars Rover's Earthly Twin." September 4, 2020. https://mars.nasa.gov/news/8749/nasa-readies-perseverance-mars-rovers-earthly-twin/.

Nath, Shyam Varan, Ann Dunkin, Mahesh Chowdhary, and Nital Patel. *Industrial Digital Transformation: Accelerate Digital Transformation with Business Optimization, AI, and Industry 4.0*. Birmingham, UK: Packt Publishing, 2020.

National Science Foundation. "Cyber-Physical Systems (CPS)." January 7, 2021. https://new.nsf.gov/funding/opportunities/cyber-physical-systems-cps.

National Science Foundation. "Cyber-Physical Systems: Enabling a Smart and Connected World." Accessed May 1, 2023. https://www.nsf.gov/news/special_reports/cyber-physical/.

National Science Foundation. "Cyber-Physical Systems (CPS): PROGRAM SOLICITATION NSF 21-551." Accessed May 21, 2023. https://www.nsf.gov/pubs/2021/nsf21551/nsf21551.htm.

Nelson, Mark. "Beyond the Buzzword: What Does Data-Driven Decision-Making Really Mean?" *Forbes*. September 23, 2022. https://www.forbes.com/sites/tableau/2022/09/23/beyond-the-buzzword-what-does-data-driven-decision-making-really-mean/?sh=7f7d2e2725d6.

Nolan, Dennis P., and Eric T. Anderson. *Applied Operational Excellence for the Oil, Gas, and Process Industries*. Waltham, MA: Gulf Professional Publishing, 2015.

Office of Nuclear Energy. "Idaho National Laboratory Demonstrates First Digital Twin of a Simulated Microreactor." July 14, 2022. https://www.energy.gov/ne/articles/idaho-national-laboratory-demonstrates-first-digital-twin-simulated-microreactor.

Office of the Under Secretary of Defense for Acquisition and Sustainment. *Agile Software Acquisition Guidebook-Best Practices & Lessons Learned from the FY18 NDAA Section 873/874 Agile Pilot Program*. Washington, D.C.: Feb. 27, 2020. See National Defense Authorization Act for Fiscal Year 2018, Pub. L. No. 115-91, §§ 873-874 (2017) (codified at 10 U.S.C. §§ 2223a note, 2302 note)· Section 874 of the NDAA for Fiscal Year 2018.

O'Shaughnessy, Robert. "NASA's Software Development Gets Complex." Federal News Network. December 14, 2022. https://federalnewsnetwork.com/technology-main/2022/12/nasas-software-development-gets-complex/.

Palucka, Tim. "NASCAR's Next Gen Race Car Proven Safe by Simulation." InfoWorld. October 19, 2022. https://www.infoworld.com/article/3676588/nascar-s-next-gen-race-car-proven-safe-by-simulation.html.

Perez, Carlotta. *Technological Revolutions and Financial Capital: The Dynamics of Bubbles and Golden Ages*. Cheltenham, UK: Edward Elgar, 2003.

Phipps, Blaine E., Michael H. Young, and Nathan G. Christensen. "Structural Optimization Helps Launch Space Payloads." *Concept to Reality.* Altair.com. Summer/Fall 2011. https://altair.com/docs/default-source/resource-library/c2r2011-structural-opt.pdf?sfvrsn=33f8a9fc_3.

Pink, Daniel. *Drive: The Surprising Truth About What Motivates Us*. Edinburgh, UK: Canongate Books, 2010.

Pollock, Roy V. H., Andy Jefferson, and Calhoun W. Wick. *The Six Disciplines of Breakthrough Learning: How to Turn Training and Development into Business Results*. Hoboken, NJ: John Wiley & Sons, 2015.

Porsche AG. "Agile Transformation: Bringing the Porsche Experience into the Digital Future with SAFe." Medium.com (wesbite). May 11, 2021. https://medium.com/next-level-german-engineering/agile-transformation-bringing-the-porsche-experience-into-the-digital-future-with-safe-bf3df9bdbd08

———. “A Journey through the history of Porsche: Why Agile Work Isn’t New to Us.” Medium.com (website). January 16, 2020. https://medium.com/next-level-german-engineering/a-journey-through-the-history-of-porsche-why-agile-work-isnt-new-to-us-9664ab30f035

Prior, Madeleine. “Launcher Successfully Tests Its 3D Printed E2 Rocket Engine in Full Thrust.” 3Dnatives. April 30, 2022. https://www.3dnatives.com/en/launcher-3d-printed-e2-rocket-engine-in-full-thrust-300420224/#!.

Rajadurai, Niroshan. “How SpaceX Develops Software.” Coders Kitchen. June 25, 2020. https://www.coderskitchen.com/spacex-software-development-and-testing/.

Ready, Douglas A., Carol Cohen, David Kiron, and Benjamin Pring. “The New Leadership Playbook for the Digital Age.” *MIT Sloan Management Review*. January 2020. https://www.cognizant.com/en_us/insights/documents/the-new-leadership-playbook-for-the-digital-age-codex5350.pdf.

Reichmann, Kelsey. “In a First for the DoD, Kubernetes Installed on U-2 Dragon Lady.” Avionics International. October 9, 2020. https://www.aviationtoday.com/2020/10/09/first-dod-kubernetes-installed-u-2-dragon-lady/.

Reinertsen, Donald G. *The Principles of Product Development Flow: Second Generation Lean Product Developmen*t. Redondo Beach, CA: Celeritas Pub, 2009.

Reiterer, Stefan H., Sinan Balci, Desheng Fu, Martin Benedikt, Andreas Soppa, and Helena Szczerbicka. “Continuous Integration for Vehicle Simulations.” *2020 25th IEEE International Conference on Emerging Technologies and Factory Automatio*n (2020): 1023–1026.

Relativity Space. “World’s Largest 3D Metal Printers: Rockets Built for the Future.” Retrieved May 12, 2023. https://www.relativityspace.com/stargate.

Riedel, Robin. “Rideshares in the Sky by 2024: Joby Aviation Bets Big on Air Taxis.” McKinsey & Company. November 19, 2021. https://www.mckinsey.com/industries/aerospace-and-defense/our-insights/rideshares-in-the-sky-by-2024-joby-aviation-bets-big-on-air-taxis.

Ries, Eric. *The Lean Startup: How Today’s Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesse*s. New York: Crown Business, 2011.

———. *The Startup Way: How Modern Companies Use Entrepreneurial Management to Transform Culture and Drive Long-Term Growth*. New York: Currency, 2017.

Rigby, Darrell, Jeff Sutherland, and Hirotaka Takeuchi. “Embracing Agile: How to Master the Process That’s Transforming Management.” *Harvard Business Review*. May 2016. https://hbr.org/2016/05/embracing-agile.

Rigby, Darrell, Jeff Sutherland, and Hirotaka Takeuchi. “The Secret History of Agile Innovation.” *Harvard Business Review*. April 20, 2016. https://hbr.org/2016/04/the-secret-history-of-agile-innovation.

Risher, Howard. “Daniel Pink’s ‘Zen of Compensation’ Is Anything But.” Compensation Café. June 8, 2012. https://www.compensationcafe.com/2012/06/daniel-pinks-zen-of-compensation-is-anything-but.html.

Rocket Lab. “Responsive Space: Accelerating the Path to Orbit with Rapid Call-Up Launch on Demand and Agile Satellite Solutions.” Accessed May 21, 2023. https://www.rocketlabusa.com/launch/responsive-space/.

Root, Al. “How Much Is SpaceX Worth Now? More Than Lockheed Martin.” Barron’s. December 13, 2022. https://www.barrons.com/articles/spacex-lockheed-martin-market-value-51670942386.

Rosenberg, Barry. “DevStar for B-21 Design, Build, Sustainment.” Breaking Defense. June 30, 2021. https://breakingdefense.com/2021/06/devstar-for-b-21-design-build-sustainment/.

Rother, Mike. *Toyota Kata: Managing People for Improvement, Adaptiveness, and Superior Results*. New York: McGraw-Hill Education, 2010.

Royce, Winston W. “Managing the Development of Large Software Systems.” *ICSE ‘87: Proceedings of the 9th International Conference on Software Engineering* (March 1981): 328–338.

Sacolick, Isaac. “3 Ways DevOps Can Support Continuous Architecture.” InfoWorld. May 30, 2022. https://www.infoworld.com/article/3662290/3-ways-devops-can-support-continuous-architecture.html.

Santos, Paula de Oliveira, and Marly Monteiro de Carvalho. “Exploring the Challenges and Benefits for Scaling Agile Project Management to Large Projects: A Review.” *Requirements Engineering* 27, no. 1 (March 2022): 117–134.

Saran, Cliff. “How Containerisation Helps VW Develop Car Software.” Computer Weekly (website). June 23, 2021. https://www.computerweekly.com/news/252502600/How-containerisation-helps-VW-develop-car-software.

Scaled Agile Framework. “Create a Lean-Agile Center of Excellence.” December 9, 2022. https://scaledagileframework.com/lace/.

Schein, Edgar H. *Organizational Culture and Leadership*. Hoboken, NJ: John Wiley & Sons, 2017.

Shepard, David, and Julia Scherb. “What Is Digital Engineering and How Is It Related to DevSecOps?” Software Engineering Institute. November 16, 2020. https://insights.sei.cmu.edu/blog/what-digital-engineering-and-how-it-related-devsecops/.

Shieber, Jonathan. “Wondering About Getting a Job at SpaceX? Elon Musk Says Innovation Is the Main Criterion.” TechCrunch. February 28, 2020. https://techcrunch.com/2020/02/28/wondering-about-getting-a-job-at-spacex-elon-musk-says-innovation-is-the-main-criterion/.

Shook, John. “How to Change a Culture: Lessons from NUMMI.” *MIT Sloan Management Review*. January 1, 2010. https://sloanreview.mit.edu/article/how-to-change-a-culture-lessons-from-nummi/.

Simon, Herbert A. *Administrative Behavior: A Study of Decision-Making Processes in Administrative Organization*. New York: The Free Press, 1997.

Simon, Herbert A., George B. Dantzig, Robin Hogarth, Charles R. Plott, Howard Raiffa, Thomas C. Schelling, Kenneth A. Shepsle, Richard Thaler, Amos Tversky, and Sidney Winter. “Decision Making and Problem Solving.” *Interfaces* 17, no. 5 (October 1987): 11–31.

Sinek, Simon. *The Infinite Game*. New York: Portfolio/Penguin, 2019.

Singh, Victor, and K. E. Willcox. “Engineering Design with Digital Thread.” *American Institute of Aeronautics and Astronautics (AIAA) Journal* 56, no. 11 (November 2018).

Skelton, Matthew, and Manuel Pais. *Team Topologies: Organizing Business and Technology Teams for Fast Flow*. Portland, OR: IT Revolution, 2019.

Smart, Jonathan. *Sooner Safer Happier: Antipatterns and Patterns for Business Agility*. Portland, OR: IT Revolution Press, 2020.

Somers, Richard J., James A. Douthwaite, David J. Wagg, Neil Walkinshaw, and Robert M. Hierons. "Digital-Twin-Based Testing for Cyber–Physical Systems: A Systematic Literature Review." *Information and Software Technology* 156 (2023).

SpaceX. "Mars & Beyond: The Road to Making Humanity Multiplanetary." Accessed May 7, 2023. https://www.spacex.com/human-spaceflight/mars/.

———. "The Future of Design." September 5, 2013. YouTube video, 3:48. https://www.youtube.com/watch?v=xNqs_S-zEBY.

Spiegel, Rob. "Simulation Brings Design Speed to NASCAR." Design News. January 31, 2021. https://www.designnews.com/design-software/simulation-brings-design-speed-nascar.

Sprovieri, John. "BMW Applies AI to Assembly." ASSEMBLY. June 29, 2022. https://www.assemblymag.com/articles/97139-bmw-applies-ai-to-assembly#:~:text=%E2%80%9CThe%20BMW%20Group%20has%20been,production%20at%20%5Bour%5D%20plants.%20%3C/Citation%3E.

Straits Research. "CubeSat Market: Information by Size (0.25U to 1U, 1U to 3U), Application (Space Observation), Subsystem (Payloads, Structures) End-Users, and Region—Forecast till 2030." Accessed May 22, 2023. https://straitsresearch.com/report/cubesat-market.

Stray, Viktoria, Bakhtawar Memon and Lucas Paruch. "Systemic Literature Review on Agile Coaching and the Role of the Agile Coach." International Conference on Product-Focused Software Process Improvement. October 2020. https://www.researchgate.net/publication/345062509_A_Systematic_Literature_Review_on_Agile_Coaching_and_the_Role_of_the_Agile_Coach.

Strickland, Ashley. "'Rail Cars' of Material Released after NASA Spacecraft Hit Asteroid." CNN. December 15, 2022. https://www.cnn.com/2022 /12/15/world/dart-mission-momentum-results-scn/index.html.

Stumpf, Rob. "At $631B, Tesla Is Now Worth More than the Next Top 6 Car Companies Combined." The Drive. December 30, 2020. https://www.thedrive.com/news/38485/at-631b-tesla-is-now-worth-more-than-the-next-top-6-car-companies-combined

Sutherland, J. J., and Joe Justice. "Agile In Military Hardware: How the SAAB Gripen became the world's most cost effective military aircraft." Scrum Inc. 2017.

Sutherland, Jeff, and J. J. Sutherland. *Scrum: The Art of Doing Twice the Work in Half the Time*. New York: Crown Business, 2014.

Szondy, David. "Lockheed Martin Develops Smart Satellite That Can Be Reprogrammed In Orbit." New Atlas. March 22, 2019. https://newatlas.com/lockheed-martin-smart-satellite/58957/.

Tao, Fei, Meng Zhang, and A. Y. C. Nee. *Digital Twin Driven Smart Manufacturing*. London: Academic Press, 2019.

Takeuchi, Hirotaka, and Ikujiro Nonaka. "The New New Product Development Game." *Harvard Business Review.* January 1986. https://hbr.org/1986/01/the-new-new-product-development-game.

Tracy, Brian. "Goals Mastery for Personal and Financial Success." BrianTracy.com. Accessed May 14, 2023. https://www.briantracy.com/files/pages/goals/mastery/lp-long.html?cmpid=2269&amp%3Bproid=2031.

Ullman, David G., and Joshua Tarbutton. "Scrum for Hardware and Systems Development." Machine Design. May 29, 2019. https://www.machinedesign.com/3d-printing-cad/article/21837829/scrum-for-hardware-and-systems-development.

US Government Accountability Office. *Agile Assessment Guide: Best Practices for Agile Adoption and Implementation*. September 2020. https://www.gao.gov/assets/gao-20-590g.pdf.

———. *F-35 Joint Strike Fighter: Assessment Needed to Address Affordability Challenges*. April 2015. https://www.gao.gov/assets/gao-15-364.pdf.

———. *Software Development: Effective Practices and Federal Challenges in Applying Agile Method*s. July 27, 2012. https://www.gao.gov/products/gao-12-681.

———. *Weapons Systems Annual Assessment: Challenges to Fielding Capabilities Faster Persist*. June 2022. https://www.gao.gov/assets/gao-22-105230.pdf.

Vacanti, Daniel S. *Actionable Agile Metrics for Predictability: An Introduction*. Self-published, March 4, 2015. Leanpub.

Verma, Pranshu. "How the 3D-Printing Community Worldwide Is Aiding Ukraine," *The Washington Post*. June 12, 2022. https://www.washingtonpost.com/technology/2022/06/12/3d-printers-ukraine-war-supplies/.

Viguié, Arnaud, and Joe Justice. "Tesla Agile Success." Agile Business Institute. Accessed May 1, 2023. https://en.abi-agile.com/tesla-agile-success/.

Wall, Mike. "SpaceX Launches 1st Test Satellites for Starlink Internet Constellation along with Spain's Paz." Space.com (website). February 22, 2018. https://www.space.com/39755-spacex-used-rocket-launches-internet-satellites.html.

Wang, Brian. "Tesla's Fully Agile Rapid Innovation." Next Big Future. December 27, 2021. https://www.nextbigfuture.com/2021/12/174206.html.

Westrum, Ron. "A Typology of Organisational Cultures." *Qual Saf Health Care* 13, no. 2 (December 2004): 22–27.

Wikipedia. "Queueing Theory." Wikipedia.com (website). Last modified April 19, 2023. https://en.wikipedia.org/wiki/Queueing_theory.

Williams, Taffy. *Think Agile: How Smart Entrepreneurs Adapt in Order to Succeed*. New York: AMACOM, 2015.

Windley, Phil. "Creating an Agile Culture through Trust and Ownership: An Interview with Pollyanna Pixton and Niel Nickolaisen." InformIT. April 10, 2014. https://www.informit.com/articles/article.aspx?p=2191027.

Witthuhn, Bri Flynn. "The Neuroscience behind Habit Change." *Forbes*. February 11, 2020. https://www.forbes.com/sites/ellevate/2020/02/11/the-neuroscience-behind-habit-change/?sh=742691976f6a.

WORKERBASE. "Benefits of Agile Manufacturing for Smart Factories." Accessed May 1, 2023. https://workerbase.com/post/benefits-of-agile-production-systems-for-smart-factories#:~:text=For%20the%20company%2C%20an%20agile,by%20improving%20productivity%20and%20OEE.

Notes

Preface

1. Johnson et al., *Industrial DevOps*.

Introduction

1. Digital.ai, *16th Annual State of Agile Report*.
2. Harvard Business Review Analytic Services, *Competitive Advantage through DevOps*.
3. Mersino, "Why Agile Is Better than Waterfall."
4. Mersino, "Why Agile Is Better than Waterfall."
5. Accialini, *Agile Manufacturing*, 26.
6. Gouré, "DoD's Move to the Cloud Is Critical to Operate at the 'Speed of Relevance.'"
7. Gouré, "DoD's Move to the Cloud Is Critical to Operate at the 'Speed of Relevance.'"
8. Benchmark International, "2022 Global Space Industry Report."
9. Harvard Business Review Analytic Services, *Competitive Advantage through DevOps*.
10. Perez, *Technological Revolutions and Financial Capital*.
11. US Government Accountability Office, *Weapons Systems Annual Assessment*.
12. Kharpal, "'Pathetic' Performance Has Left U.S. 'Well Behind' China in 5G Race, Ex-Google CEO Eric Schmidt Says."
13. Feldscher, "China Could Overtake US in Space without 'Urgent Action,' Warns New Pentagon Report."
14. Ries, *The Lean Startup*.
15. Donahue, "How to Keep Pace with Digital Transformation and Avoid Becoming Obsolete."
16. Harter, "Is Quiet Quitting Real?"
17. Mersino, "Why Agile Is Better than Waterfall."
18. Mersino, "Why Agile Is Better than Waterfall."
19. Porsche AG, "A Journey through the History of Porsche."
20. Porsche AG, "Agile Transformation."
21. Afifi-Sabet, "How BMW Embraced Agile to Hit New Speeds."
22. Afifi-Sabet, "How BMW Embraced Agile to Hit New Speeds."
23. Kersten, *Project to Produc*t, 190.
24. Field, "Tesla Has Applied Agile Software Development to Automotive Manufacturing."
25. Viguié and Justice, "Tesla Agile Success."
26. Wang, "Tesla's Fully Agile Rapid Innovation."
27. Wang, "Tesla's Fully Agile Rapid Innovation."
28. Wang, "Tesla's Fully Agile Rapid Innovation."
29. Johnson et al., *Industrial DevOps*.
30. Deming, *Out of the Crisis*.

Chapter 1

1. Nath et al., *Industrial Digital Transformation*.
2. Ross, as quoted in: Carr, "Digital Transformation Leaders' Secret."
3. LaBerge et al., "How COVID-19 Has Pushed Companies over the Technology Tipping Point—and Transformed Business Forever."
4. "Cyber-Physical Systems (CPS)," National Science Foundation.
5. "Cyber-Physical Systems," National Science Foundation.
6. Mahmoud and Xia, "Chapter 13—Secure Estimation Subject to Cyber Stochastic Attacks," in *Cloud Control Systems*, edited by Ison and Budd, 373–404.
7. Musk, as quoted in: Shieber, "Wondering About Getting a Job at SpaceX?"
8. Viguié and Justice, "Tesla Agile Success."
9. Viguié and Justice, "Tesla Agile Success."
10. Stumpf, "At $631B, Tesla Is Now Worth More than the Next Top 6 Car Companies Combined."
11. Berg, "SpaceX's Use of Agile Methods."
12. Root, "How Much Is SpaceX Worth Now? "
13. Howard, "What Is Agile Aerospace?"
14. Ferreira, "Warming to Her Task."
15. Furuhjelm et al., "Owning the Sky with Agile."
16. O'Shaughnessy, "NASA's Software Development Gets Complex."
17. US Government Accountability Office, *Weapons Systems Annual Assessmen*t.

Chapter 2

1. Mochari, "Why Half of the S&P 500 Companies Will Be Replaced in the Next Decade."
2. Harvard Business Review Analytic Services, *Competitive Advantage*.
3. Wang, "Rapid Innovation."
4. Gruver and Mouser, *Leading the Transformation*, 15.
5. de Oliveira Santos and Monteiro de Carvalho, "Exploring the Challenges and Benefits for Scaling Agile Project Management to Large Projects," 117–134.
6. Klender "Tesla Makes $9,500 per Car, Eight Times as Much as Toyota."
7. Smart et al., *Sooner Safer Happier*.
8. Harvard Business Review Analytic Services, *Competitive Advantage*.
9. Sutherland and Sutherland, *Scrum*.
10. Rigby, Sutherland, and Takeuchi, "Embracing Agile."
11. Rigby, Sutherland, and Takeuchi, "Embracing Agile."
12. Jurkiewicz et al., *Whitepaper: Employee Engagement*.
13. Devalekar, "Why High Employee Engagement Results in Accelerated Revenue Growth."
14. Rigby, Sutherland, and Takeuchi, "Embracing Agile."
15. "Benefits of Agile Manufacturing for Smart Factories," WORKERBASE.

Chapter 3

1. Johnson et al., "Overcoming Barriers to Industrial DevOps."
2. "Principles behind the Agile Manifesto," The Agile Manifesto.
3. "Manifesto for Agile Software Development," The Agile Manifesto.
4. Marquet, *Turn the Ship Around*, 161.

5. "Principles behind the Agile Manifesto," The Agile Manifesto.
6. Comella-Dorda et al., "Revisiting Agile Teams after an Abrupt Shift to Remote."
7. Cleland-Huang et al., "Visualizing Change in Agile Safety-Critical Systems," 43–51.
8. Johnson et al., "Overcoming Barriers to Industrial DevOps."

Chapter 4

1. Edwards et al., *Making Matrixed Organizations Successful with DevOps*.
2. Skelton and Pais, *Team Topologies*.
3. "Value Streams," Lean Enterprise Institute.
4. Justice, personal communication with the authors, 2022.
5. Windley, "Creating an Agile Culture through Trust and Ownership."
6. Ullman and Tarbutton, "Scrum for Hardware and Systems Development"; Furuhjelm et al., "Owning the Sky"; Sutherland and Justice, "Agile in Military Hardware."
7. Furuhjelm et al., "Owning the Sky."
8. Anderson, "Code on the Road."
9. Anderson, "Code on the Road."
10. Kersten, *Project to Product*, 83.
11. Kersten, *Project to Product*, 78.
12. Anderson, "Code on the Road."

Chapter 5

1. Royce, "Managing the Development of Large Software Systems," 328–338.
2. Royce, "Managing the Development of Large Software Systems," 328–338.
3. Morris, *Storm on the Horizon*.
4. Reinertsen, *The Principles of Product Development Flow*.
5. "Mars & Beyond," SpaceX.
6. "We Are Bosch," Bosch.
7. "The Chevron Way," Chevron.
8. "Apple Inc. Form 10-K for the Fiscal Year Ended September 30, 2017," Apple, Inc.
9. "Our Missions and Values," NASA.
10. "Who We Are," Amazon
11. Ries, *The Startup Way*.
12. Wall, "SpaceX Launches 1st Test Satellites for Starlink Internet Constellation along with Spain's Paz."
13. Lofgren, "How Saab Uses Agile Principles to Develop the Gripen Fighter."
14. Lofgren, "How Saab Uses Agile Principles to Develop the Gripen Fighter."
15. US Government Accountability Office, *F-35 Joint Strike Fighter*; Lofgren, "How Saab Uses Agile Principles to Develop the Gripen Fighter"; Furuhjelm et al., "Owning the Sky."
16. Lofgren, "How Saab Uses Agile Principles to Develop the Gripen Fighter."

Chapter 6

1. Nelson, "Beyond the Buzzword."
2. Reinertsen, *Principles of Product Development Flow*.
3. Lofgren, "How Saab Uses Agile Principles to Develop the Gripen Fighter."
4. Sutherland and Justice, "Agile in Military Hardware."
5. Singh and Willcox, "Engineering Design with Digital Thread."
6. Cleland-Huang et al., "Visualizing Change."

7. Somers et al., "Digital-Twin-Based Testing for Cyber-Physical Systems."
8. Joe Justice, personal communication with the authors, 2022.
9. Augustine, Cuellar, and Scheere, *From PMO to VMO*.
10. Doerr, *Measure What Matters*.
11. Kersten, *Project to Product*, 111.
12. "What Is Quality," ISO Update.
13. "Flow metrics," Flow Framework.
14. Kersten, *Project to Product*.
15. Lean Enterprise Institute, "Cycle Time."
16. Landau, "What Is Lead Time?
17. Lean Enterprise Institute, "Cycle Time."
18. Vacanti, *Actionable Agile Metrics for Predictability*.
19. Kersten, *Project to Product*, 138.
20. Datta, "Best of 2022: How DORA Metrics Can Measure and Improve Performance."
21. Forsgren, Humble, and Kim, *Accelerate*, 17.
22. Somers et al., "Digital-Twin-Based Testing."
23. Spiegel, "Simulation Brings Design Speed to NASCAR"; Palucka, "NASCAR's Next Gen Race Car Proven Safe by Simulation"; "Simulation Advances NASCAR Race Car Development," Digital Engineering 24/7.

Chapter 7

1. Szondy, "Lockheed Martin Develops Smart Satellite That Can Be Reprogrammed in Orbit."
2. Reichmann, "In a First for the DoD, Kubernetes Installed on U-2 Dragon Lady."
3. "Formula 1 Redesigns Car for Closer Racing and More Exciting Fan Experience by Using AWS HPC Solutions," Amazon Web Services.
4. Howell, "8 Ways That SpaceX Has Transformed Spaceflight."
5. Lambert, "First Look at Tesla's New Structural Battery Pack That Will Power Its Future Electric Cars."
6. Verma, "How the 3D-Printing Community Worldwide Is Aiding Ukraine."
7. "World's Largest 3D Metal Printers," Relativity Space.
8. Blinde, "Lockheed Martin Completes First LM 400 Multi-Mission Spacecraft."
9. Saran, "How Containerisation Helps VW Develop Car Software."
10. Muruganandham, "Cloud cCmputing: Ratcheting the Satellite Industry Forward."
11. "3 Toyotas with the Lowest 10-Year Maintenance Costs," MotorBiscuit.
12. Daily and Peterson, "Predictive Maintenance: How Big Data Analysis Can Improve Maintenance."
13. Fritzsch et al., "Adopting Microservices and DevOps in the Cyber-Physical Systems Domain," 790–810.
14. Sacolick, "3 Ways DevOps Can Support Continuous Architecture."
15. Sacolick, "3 Ways DevOps Can Support Continuous Architecture."
16. Rosenberg, "DevStar for B-21 Design, Build, Sustainment."
17. Sprovieri, "BMW Applies AI to Assembly."
18. "From Sample To Results: In-Space Data Analysis Enables Quicker Data Return," ISS National Laboratory.
19. Khan, "Evolution of Mars Rovers in the Past 25 Years."
20. Howard, "What Is Agile Aerospace?"
21. Moorhead, "The Past and Present Are Telling of the Future, So What's Next for Planet?"

22. Riedel, “Rideshares in the Sky by 2024.”
23. Bellan, “Joby Aviation’s Contract with US Air Force Expands to Include Marines.”
24. Kolodny, “Joby Aviation Can’t Hit Production Targets on Time, According to Short Sellers’ Report.”

Chapter 8

1. Brooks, “Why Did the Wright Brothers Succeed When Others Failed?”
2. Brooks, “Why Did the Wright Brothers SucceedWhen Others Failed?”
3. Harvard Business Review Analytic Services, *Competitive Advantage*.
4. Simon, *Administrative Behavior*.
5. Simon et al., “Decision Making and Problem Solving,” 11–31.
6. Kennedy, *Product Development for the Lean Enterpris*e, 22.
7. Kennedy, *Product Development for the Lean Enterprise*.
8. Reinertsen, *Principles of Product Development Flow*.
9. Wikipedia, “Queueing Theory.”
10. Johnson et al., “Overcoming Barriers to Industrial DevOps.”
11. Bloomberg Technology, “SpaceX Nails Landing of Reusable Rocket on Land.”
12. Carlos, “SpaceX.”
13. Carlos, “SpaceX.”
14. Prior, “Launcher Successfully Tests Its 3D Printed E2 Rocket Engine in Full Thrust.”
15. Phipps, Young, and Christensen, “Structural Optimization Helps Launch Space Payloads.”

Chapter 10

1. “World’s Largest 3D Metal Printers,” Relativity Space.
2. Kim et al., *The DevOps Handbook*, 153–154.
3. Humble and Farley, *Continuous Delivery*, 58–59.
4. Kowzan and Pietrzak, “Continuous Integration in Validation of Modern, Complex, Embedded Systems,” 160–164.
5. Kowzan and Pietrzak, “Continuous Integration in Validation of Modern, Complex, Embedded Systems.”
6. Tao, Zhang, and Nee, *Digital Twin Driven Smart Manufacturing*.
7. Kowzan and Pietrzak, “Continuous Integration in Validation of Modern, Complex, Embedded Systems.”
8. Reiterer et al., “Continuous Integration for Vehicle Simulations,” 1023–1026.
9. Automotive World, “Bosch: Did You Know. . . .”

Chapter 11

1. European Space Agency, “No 33–1996: Ariane 501.”
2. Ismail, “AI Solutions Required for Fast-Paced Application Development?”
3. Rajadurai, “How SpaceX Develops Software.”
4. Rajadurai, “How SpaceX Develops Software.”
5. Europeans Space Agency, “Galileo Clock Anomalies under Investigation.”
6. Heistand et al., “DevOps for Spacecraft Flight Software,” 1–16.
7. Strickland, “‘Rail Cars’ of Material Released after NASA Spacecraft Hit Asteroid.”
8. International Organization for Standardization, “ISO 26262-1:2011 Road Vehicles — Functional Safety — Part 1.”

9. Simon, “Massive Autonomous Vehicle Sensor Data: What Does It Mean?,” .
10. Gartner, “Use Test-First Development Processes to Jump-Start Your SDLC.”
11. Forbes Technology Council, “11 Benefits of Behavior-Driven Product Development.”
12. “NASA Readies Perseverance Mars Rover’s Earthly Twin,” NASA Science Mars Exploration.
13. “Satellite Testing: Best Practices for Application Performance Testing Over Satellite,” Apposite Technologies.
14. “Idaho National Laboratory Demonstrates First Digital Twin of a Simulated Microreactor,” Office of Nuclear Energy.

Chapter 12

1. Dweck, *Mindset*.
2. Westrum, “A Typology of Organisational Cultures.”
3. Risher, “Daniel Pink’s ‘Zen of Compensation’ Is Anything But.”
4. Tracy, “Goals Mastery for Personal and Financial Success.”
5. Ries, *Lean Startup*.
6. Schein, *Organizational Culture and Leadership*.
7. Grant, *Think Again*.
8. Schein, *Organizational Culture*, 334.
9. Kohn, “Why Incentive Plans Cannot Work.”
10. Pink, *Drive*.
11. Kotter, “Leading Change.”
12. Kohn, “Why Incentive Plans.”
13. Johnson et al., *Building Industrial DevOps Stickiness*.
14. Edmondson, “Strategies for Learning from Failure.”
15. Anderson, *Learning to Lead, Leading to Learn*, 348.
16. Edmondson, “Strategies for Learning from Failure.”
17. Pollock, Jefferson, and Wick, *The Six Disciplines of Breakthrough Learning*.
18. Pollock, Jefferson, and Wick, *The Six Disciplines of Breakthrough Learning*.
19. Deutschman, “Change or Die.”
20. US Government Accountability Office, *Agile Assessment Guide*.
21. Stray, Memon, and Paruch, “Systemic Literature Review on Agile Coaching and the Role of the Agile Coach.”
22. Bodell, “Why T-Shaped Teams Are the Future of Work.”
23. Hamdi et al., “Impact of T-Shaped Skill and Top Management Support on Innovation Speed.”
24. Schein, *Organizational Culture*.
25. Pollock, Jefferson, and Wick, *The Six Disciplines of Breakthrough Learning*.
26. Groysberg et al., “The Leader’s Guide to Corporate Culture.”
27. Dweck, *Mindset*, 127.
28. Groysberg et al., “Corporate Culture.”
29. Crook et al., *Applied Industrial DevOps 2.0*.
30. Shook, “How to Change a Culture.”
31. Shook, “How to Change a Culture.”

Chapter 13

1. Ready et al., “The New Leadership Playbook for the Digital Age.”

2. "Hoshin Planning," Gemba Academy.
3. Doerr, *Measure What Matters*.
4. Augustine, Cuellar, and Scheere, *From PMO to VMO*, 37.
5. Nolan and Anderson, *Applied Operational Excellence for the Oil, Gas, and Process Industries*.
6. Crook et al., *Applied Industrial DevOps 2.0*.
7. Rother, *Toyota Kata*.
8. "Responsive Space: Accelerating the Path to Orbit with Rapid Call-Up Launch on Demand and Agile Satellite Solutions," Rocket Lab.

Chapter 14

1. International Center for Clinical Excellence, "The Backwards Bicycle."
2. Witthuhn, "The Neuroscience behind Habit Change."
3. Digital.ai, *15th Annual State of Agile Report*.
4. US Government Accountability Office, *Weapons Systems Annual Assessment*.
5. McDermott, "Prospect Theory."
6. Maxwell, *Leadershift*, 6.
7. Dikert, Paasivaara, and Lassenius, "Challenges and Success Factors for Large-Scale Agile Transformations."
8. Kalenda, Hyna, and Rossi, "Scaling Agile in Large Organizations."
9. Deming, *Out of the Crisis*.
10. Williams, *Think Agile*, 6.
11. Grant, *Think Again*, 255.
12. Digital.ai, *15th Annual State of Agile Report*.
13. US Government Accountability Office, *Weapons Systems Annual Assessment*.
14. US Government Accountability Office, *Weapons Systems Annual Assessment*.
15. Digital.ai, *15th Annual State of Agile Repor*t.
16. Dikert, Paasivaara, and Lassenius. "Challenges and Success Factors for Large-Scale Agile Transformations."
17. US Government Accountability Office, *Software Development*.
18. Dikert, Paasivaara, and Lassenius. "Challenges and Success Factors for Large-Scale Agile Transformations."
19. Kotter, "Leading Change."
20. "Create a Lean-Agile Center of Excellence," Scaled Agile Framework.
21. Kontoghiorghes, Awbre, and Feurig, "Examining the Relationship between Learning Organization Characteristics and Change Adaptation, Innovation, and Organization Performance."
22. Digital.ai, *15th Annual State of Agile Report*.
23. Errick, "Senate Committee Approves AGILE Procurement Act for IT and Communications Tech."
24. "Cyber-Physical Systems (CPS): PROGRAM SOLICITATION NSF 21-551," National Science Foundation.
25. Office of the Under Secretary of Defense for Acquisition and Sustainment, *Agile Software Acquisition Guidebook*.

Conclusion

1. Sinek, *The Infinite Game*.
2. Smart et al., *Sooner Safer Happier*.

Appendix A

1. “The Cost of Building and Launching a Satellite,” GlobalCom.
2. “CubeSat Market,” Straits Research.

Appendix B

1. “What is Lean?” Lean Enterprise Institute.
2. Lawton, “The History of Lean – Part 1.”
3. Ford, “The Moving Assembly Line.”
4. Ford, “The Moving Assembly Line.”
5. Hessing, “History of Lean.”
6. Ries, *The Lean Startup*.
7. Brenton, “What Is Value Stream Management?”
8. Rigby, Sutherland, and Takeuchi, “The Secret History of Agile Innovation.”
9. Takeuchi and Nonaka, “The New New Product Development Game.”
10. Fowler and Highsmith, “The Agile Manifesto.”
11. Agile Alliance, “Subway Map to Agile Practices.”
12. IEEE Standards Association, “IEEE 2675-2021.”
13. INCOSE, “Systems and SE Definitions.”
14. Biggs, “What Is a Business Architect and How Do You Become One?”
15. Ferguson, “Application Architecture.”
16. Buchanan, “Wicked Problems in Design Thinking.”
17. Liedtka, “Why Design Thinking Works.”
18. Shepard and Scherb, “What Is Digital Engineering and How Is It Related to DevSecOps?”
19. Accialini, *Agile Manufacturin*g, IX.
20. Accialini, *Agile Manufacturing*, XIX.
21. Kumar and Prasad, “Chapter 2 - Basic Principles of Additive Manufacturing.”

Index

Acknowledgments

We have been on this journey for many years. The first time we presented together was on a panel at an Agile conference in Washington, DC, about ten years ago. The stories and experiences (and even some of the data) we shared were so similar we decided to join forces. Our thinking was that by working together and working with others across industry, it could drive us all toward better ways of working and delivering value faster to our customers. We have worked with many amazing people from across industry who have, in a variety of different ways, contributed to the knowledge and experiences captured in this book.

Thank you to our customers and teams: Many of these experiences are a result of the customers we have served and the teams we have worked with in support of those missions. We appreciate all we learned from those experiences, including the importance of collaboration, service, commitment, and delivering value to meet mission needs.

Thank you to Gene Kim: With great appreciation, we would like to thank Gene Kim. He provided us with a platform on many occasions to share our ideas. He and those in the DevOps Enterprise community have listened and provided input as we have shaped the principles of Industrial DevOps since 2018. This book is a direct result of his support.

Thank you to Leah Brown: We also want to acknowledge and offer special thanks to Leah Brown, Managing Editor at IT Revolution. She had the challenge of keeping us focused and on track! She met with us nearly every week as we wrote this book. Her time and expertise have been invaluable.

Thank you to Anna Noak: Thank you for getting us started! We appreciate the many discussions we had during the DevOps Enterprise Summit and other events as we formulated ideas for sharing our learning and experiences with others.

Thank you to the early contributors: We also want to recognize the many contributors who have specifically helped shape the principles of Industrial DevOps over the years. Several articles on Industrial DevOps were written prior to this book. We appreciate the time, energy, and expertise of these individuals. Co-authors and reviewers of the Industrial DevOps papers are: Josh Atwell, Matt Aizcorbe, Dr. Jeff Boleng, Deborah Brey, Adrian Cockcroft, Josh Corman, Joy Crook, Steve Farley, Ben Grinnell, George Haley, Steve Holt, Diane LaFortune, Dean Leffingwell, Vincent Lussenburg, Dr. Mik Kersten, Harry Koehnemann, Dr. Stephen Magill, Dr. Steve Mayner, Michael McKay, Avigail Ofer, Dantar Oosterwal, Jeffrey Shupack, Dr. Steve Spear, Robert Stroud, Cat Swetel, Anders Wallgren, Hasan Yasar, Simon Wardley, Dr. Ron Westrum, and Eileen Wrubel.

Thank you to our book review team: Special thank you to our book reviewers! Our reviewers come from a variety of backgrounds, experiences, and industries. Their insights, recommendations, and encouragement have been greatly valued. Their honest and open feedback helped us to shape the content and provide clarity where it was needed. Thank you so much for staying on this journey with us. The book reviewers include Nicola Accialini, Sanjiv Augustine, Dawn Beyer, Jeff Boleng, Nathan G. Christensen, Gabriela (Gabby) Cole, Braxton Cook, Nathaniel Crews, Carol Erikson, Jennifer Fawcett, Sandra J. Forney (DEng), Kyle Fox, Marshall Guillory, Laura Hart, Rick Hefner (PhD), Luke Hohmann, Kelli Houston, Judy Johnson, James (Jim) Judd, Joe Justice, Harry Koehnemann, Gordon Kranz, Eugene Lai, Tom McDermott, Edward Moshinsky, Andrew Olguin, Larri A. Rosser, Isaac Sacolick, Robert Scheurer, Steve Simske, Michael P. Trombley, Valerie Underwood, John Wetsch, Christopher D. Williams, and Hasan Yasar.

Thank you to our families: And we, of course, want to say thank you to our families, who have supported us along the way. Thank you so much for the encouragement and patience as we worked diligently month after month, year after year on this journey. This journey is so much better because of you.

About the Authors

Dr. Suzette Johnson

My initial experience with Agile-related practices began in the 1990s with product development for an innovative and cutting-edge technology startup company during the dot-com era. While we knew nothing about Agile frameworks, we did understand the importance of delivering value to the customer through short development cycles.

For the majority of my career I have worked for Northrop Grumman Corporation, a global aerospace, defense, and security company. While working for Northrop Grumman, my experiences with Scrum and eXtreme programming officially started in 2005 when working on a large data-centric program. I have been actively promoting these principles ever since.

As the Lean Agile transformation lead, I launched the Northrop Grumman Lean Agile Community of Practice and the Lean Agile Center of Excellence, providing resources to a workforce of over 95,000 people. Over the years, I have had the privilege to support over one hundred enterprise, federal, and Department of Defense initiatives in their adoption of Lean Agile and DevOps for improved business agility.

In my current role as a Northrop Grumman Fellow and Technical Fellow Emeritus, I am focused on the adoption of Industrial DevOps principles within the space sector.

I am an active participant in the National Defense Industrial Association (NDIA) Systems Engineering Division, NDIA ADAPT (Agile Delivery for Agencies, Programs, and Teams) working group, and the International Council on Systems Engineering (INCOSE). I also serve as a volunteer in K-12 education to share the fun and excitement of Science, Technology, Engineering, and Math (STEM) to inspire and grow our future leaders.

I received a Doctorate of Management at the University of Maryland with a dissertation focused on investigating the impact of leadership styles on software project outcomes in traditional and Agile engineering environments. I currently reside in Maryland with my family and our two lively Jack Russells. I look forward to continuing this journey as we advance and evolve in the digital age and build better systems faster.

Robin Yeman

I spent twenty-six years working at Lockheed Martin in various roles leading up to senior technical fellow building large systems including everything from submarines to satellites. I led the Agile community of practice supporting a workforce of 120,000 people. My initial experience with Lean practices began in the late ’90s. In 2002, I had the opportunity to lead my first Agile program with multiple Scrum teams. After I had a couple months of experience, I was hooked and never turned back. I both led and supported Agile transformations for intelligence, federal, and Department of Defense organizations over the next two decades, and each one was more exciting and challenging than the last. In 2012, I had the opportunity to extend our Agile practices into DevOps, which added extensive automation and tightened our feedback loops, providing even larger results.

I have consulted for a range of Fortune 500 companies in highly regulated environments, enabling them to achieve the same results we experienced at Lockheed Martin. I engage in everything from automotive, pharmaceuticals, and energy to reimagining legacy to modern solutions using all of the tools in my toolbox, including Agile, DevOps, Lean, digital engineering, systems theory, design thinking, and more.

Currently, I am the Space Domain Lead at the Software Engineering Institute at Carnegie Mellon University.

I am and always will be a continuous learner. My education includes a bachelor’s degree from Syracuse University in Computer Information Systems and a master’s degree from Rensselaer Polytechnic Institute in Software Engineering, and I’m currently pursuing a PhD in Systems Engineering at Colorado State University, where I am working on my contribution to demonstrate empirical data of the benefits of implementing Agile and DevOps for safety-critical cyber-physical systems.