Altium

Modified Agile for Electronics Development:

A Smarter Path to High-Value Solutions

Dorian Simpson
MAHD Framework
Founder & Lead Consultant

Vishal Sheth
MAHD Framework Partner
& Lead Consultant

Lena Węglarz
Altium Expert

Contents

Is Agile Right for Electronics?

Multiple Forces Require a More Agile Approach to Electronics

While agile methods are gaining traction for physical products, most organizations, including those focused on electronics development, still rely heavily on traditional waterfall processes such as phase-gate systems, even with their well-known limitations. Traditional approaches often follow these steps:

1. A business case is created.
2. A Product Manager defines the solution.
3. The scope and schedule are negotiated with R&D teams.
4. This agreement becomes an implied contract with an expectation of delivering a specific product by a set date.

While this methodology appears reasonable and gives stakeholders a (false) sense of security, it has fundamental weaknesses based on front-end assumptions, such as:

1. Customer and market needs are clearly known and remain stable.
2. Written requirements perfectly capture both customer needs and the optimal solution.
3. The R&D team can plan and execute development with precision.

Teams can take months to work through the negotiations, but unfortunately, these assumptions rarely hold true, and much of that time and energy is wasted.

Why Traditional Methods Fall Short for Electronics Development

Most electronics projects involve complex dependencies, cross-functional collaboration, and evolving requirements. Traditional development methods struggle in this environment due to:

1. **Long Development Cycles**
   - Electronics development often spans months or even years, making it difficult to plan everything upfront.
   - Market demands and customer expectations evolve faster than the rigid waterfall process can adapt.
2. **Difficulty in Capturing Real Customer Needs**
   - Waterfall processes rely on extensive front-loaded requirements, but these are often based on assumptions rather than real-world feedback.
   - Agile allows for iterative validation, ensuring that products better align with customer expectations.
3. **Cross-Disciplinary Dependencies**
   - Unlike software, hardware development requires mechanical, electrical, and firmware teams to work in sync. Traditional methods create silos between these teams, slowing collaboration and increasing misalignment.
   - Changes in one discipline (e.g., PCB layout) can impact others (e.g., mechanical enclosure design).
4. **High Risk of Late-Stage Changes**
   - Because architecture and feature decisions are often locked in early, unexpected issues arise late in development, causing costly redesigns.
   - Changes to supply chain availability, compliance regulations, or performance requirements can disrupt timelines.

While phased NPD systems seem logical, they are not designed to deliver the most valuable solutions quickly. The results for most organizations are often missed deadlines, wasted effort, and suboptimal value.

Electronics Needs for a More Adaptive Approach

To overcome these challenges, electronics teams need a more flexible, iterative, and customer-driven approach—one that aligns with agile principles while accounting for the unique needs of hardware development.

An Agile for Software Refresher

Agile methods gained popularity in software development as a response to the inefficiencies of waterfall processes, where negotiating ever-changing "requirements" often took as long as writing code. The Agile Manifesto introduced four core principles:

- Individuals and interactions over processes and tools
- Working solutions over extensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a rigid plan

This shift led to frameworks like Scrum and SAFe™, with user stories, product owners, and backlogs becoming standard. While agile has worked well for software, where strong governance is less critical, directly applying software-driven agile tactics to hardware development often leads to frustration—whether for simple components or complex electronic systems.

The "Faux Agile" Problem for Electronics Teams

Electronics teams that attempt to apply software-based agile methods often face frustration and limited success. While principles like collaboration, short cycles, and team empowerment make sense, software-driven agile tactics—such as writing user stories for every task, building functional prototypes every iteration, and maintaining a single prioritized backlog—simply do not fit hardware development. Developing a circuit as a "user story" is impractical, frequent prototypes are too costly, and a single backlog cannot manage diverse dependencies. Many teams abandon the attempt, concluding that "agile doesn't work for hardware."

The issue lies in fundamental differences between software and hardware. Hardware teams operate under stricter constraints, complex interdependencies, and different progress metrics. Software teams can reprioritize features at any time—if something is not working, they change it in the next iteration. In hardware, decisions are governed by the classic "triple constraint" of schedule, scope, and resources. Once a major decision is made, it impacts the entire organization, from the CEO to customer support.

Unlike software teams, hardware teams cannot simply shift features to the next release. Each decision affects cost, time-to-market, and product value. These choices are often second-guessed or overridden by executives responsible for sales, margins, and market success, yet unfamiliar with project intricacies.

Expecting software-based agile tactics to work for hardware is like trying to tune a guitar with a wrench or a sculpting marble with a butter knife—they are the wrong tools for the jobs, and the results won't be great. Table 1 highlights key differences between software and hardware, underscoring why agile approaches must be adapted to deliver real benefits.

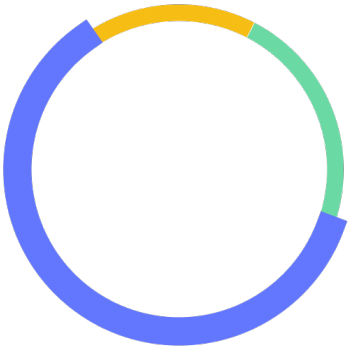
Table 1: Hardware Vs. Software Project Characteristics

PROJECT ATTRIBUTE	SOFTWARE	HARDWARE
Teams	Various software skills, mostly dedicated	Cross-discipline/SME's, many shared across projects
Constraints	Few – platform, resources	Many – cost, schedule, resources Physics, standards, etc.
Decisions	Which feature is next? (Can change next iteration!)	What is the optimal tradeoff? (We must live with this!)
Iteration Outcome	Releasable code	Progress
Governance	Priority of tasks in the backlog	Go/kill? Right product? On track?
Prototypes	Working software	Demonstrable output
Progress	# of features delivered	Remaining work to launch
Project Selection	Not necessary. The solution lives on.	Which product? Which market? What ROI?

Several Paths Have Led to Various Levels of Agile Success

Successful hardware and electronics teams recognize the limitations of software-based agile and typically adopt agile in one of three ways, each offering various levels of benefit:

Adopt Agile Basics, Ignore the Rest	The most common approach is to retain a waterfall process while incorporating short execution cycles—such as two- or four-week sprints or longer SAFe Program Increments. Often called «water-agile-fall» or «faux agile,» this method improves transparency and collaboration but provides limited gains in speed and efficiency since fundamental constraints of traditional processes remain unchanged.
Modify Agile Through Trial and Error	Some teams start with Scrum or SAFe™, then experiment with modifications—increasing prototyping frequency or introducing customer feedback loops while minimizing software-based agile rituals. However, they often still rely on front-loaded requirements and Gantt charts, limiting the full potential of agile. This approach provides more benefits than «faux agile» but still limits the real potential of agile.
Implement an Agile Framework Designed for Hardware	The most successful teams use an agile framework built for hardware, such as the Modified Agile for Hardware Development (MAHD) Framework™. Rather than starting with detailed requirements, teams align on strategic intent and prioritized customer outcomes, then collaboratively develop the path forward. Iterative cycles focus not just on execution but also on learning milestones and optimizing solutions within constraints of time, cost, and resources. Based on a survey of over 200 teams, those who implement the MAHD Framework report up to 50% faster development, improved collaboration, and increased solution value. Notably, teams reported greater speed benefits when agile methods were introduced earlier in the project. While each approach offers some agile benefits, a hardware-specific framework maximizes agility's full potential.



Speed Benefits of MAHD Projects

From initial approval to first customer shipment, what change in time-to-market have you experienced, or do you anticipate, for the projects developed using the MAHD Framework? (Estimate in increments of 5%)

- 20 Projects (40-50% Faster)
- 30 Projects (<=15% Faster)
- 62 Projects (20-30% Faster)

Source: MAHD Framework Survey, October 2024

Electronics Development Has Additional Challenges

Before diving into key agile concepts for hardware teams, it is important to understand the unique challenges of electronics development. While electronics share common hardware constraints that make software-based agile methods difficult to apply directly, additional factors come into play depending on where a solution fits within the value chain.

Imagine developing a satellite communication system where electronics are essential at every level, from integrated circuits (ICs) and PCBs to devices and full systems. Depending on your role, you may be developing:

1. Sellable electronic components or devices for external customers, or
2. Electronics as an internal system enabler integrated into larger solutions before reaching end customers.

For example, a logic or control IC developer may have an internal PCB development team as their customer. That PCB team, in turn, may serve a device team, which then integrates the technology into a full system. At any stage, these solutions may also be sold as standalone products to external customers.

Table 2: Two Unique Categories of Electronic Development

CATEGORY	SELLABLE ELECTRONIC COMPONENTS OR DEVICES	ELECTRONICS AS A PRODUCT OR SYSTEM ENABLER
Description	Electronics developed for sale as standalone products.	Electronics integrated into internal solutions.
Primary Customers	External customers such as suppliers, distribution partners, and device/system manufacturers.	Internal teams such as R&D, system developers, and solution engineers.
Business Model	Profit, market timing, and competitive positioning are key drivers.	Optimized for cost, internal schedules, and solution enablement.

The rest of this whitepaper explores how agile must be adapted for electronics development to accelerate timelines, maximize customer value, and improve predictability. We will also address how to approach electronics as a market-driven product vs. an internal system enabler and explore key enablers for agile success in electronics development.

The Overview of the Modified Agile for Hardware Development (MAHD) Framework

Electronics teams adopting hardware-specific agile methods see significant gains in speed, value, and predictability. The MAHD Framework (Figure 2) integrates user stories, prioritized backlogs, and iterative cycles while introducing key adaptations for hardware development to maximize the benefits of agile methods. First, let us consider how projects are selected.

Project Selection and Governance

Project selection is often overlooked in agile discussions, as software development typically focuses on feature updates rather than selecting entirely new projects. In contrast, hardware and electronics portfolios require careful evaluation, resource allocation, and team ramp-ups as projects progress. Agile methods should also support these governance decisions by improving flexibility and alignment with business needs.

While agile portfolio management is beyond the scope of this whitepaper, one key consideration for electronics teams is whether a solution is developed for external markets or built for internal integration within a larger system. Table 3 summarizes the different decision-making approaches for each business model.

Table 3: Business Decisions Vary by Electronics Position in the Value Chain

	SELLABLE ELECTRONIC COMPONENTS OR DEVICES	ELECTRONICS AS A PRODUCT OR SYSTEM ENABLER
Portfolio Approach	Product portfolio-based decisions	Value stream-based decisions
Key Questions	1. Which new products should we develop based on the roadmap? 2. Which existing products should be upgraded or cost-reduced? 3. What is the optimal level of resourcing for a segment, product line, or product?	1. What is the system roadmap? 2. What enabling electronic capabilities must be developed? 3. How should resources be allocated to optimize system value across components?

Understanding how projects are selected and governed helps ensure agile tactics are applied effectively within the broader hardware development process. From here, there are two fundamental areas of agile tactics that should be modified to obtain the benefits of agile principles:

1. The essential need to initiate projects for agile success;
2. How the project evolves through iterative development cycles.

The MAHD Framework addresses both.

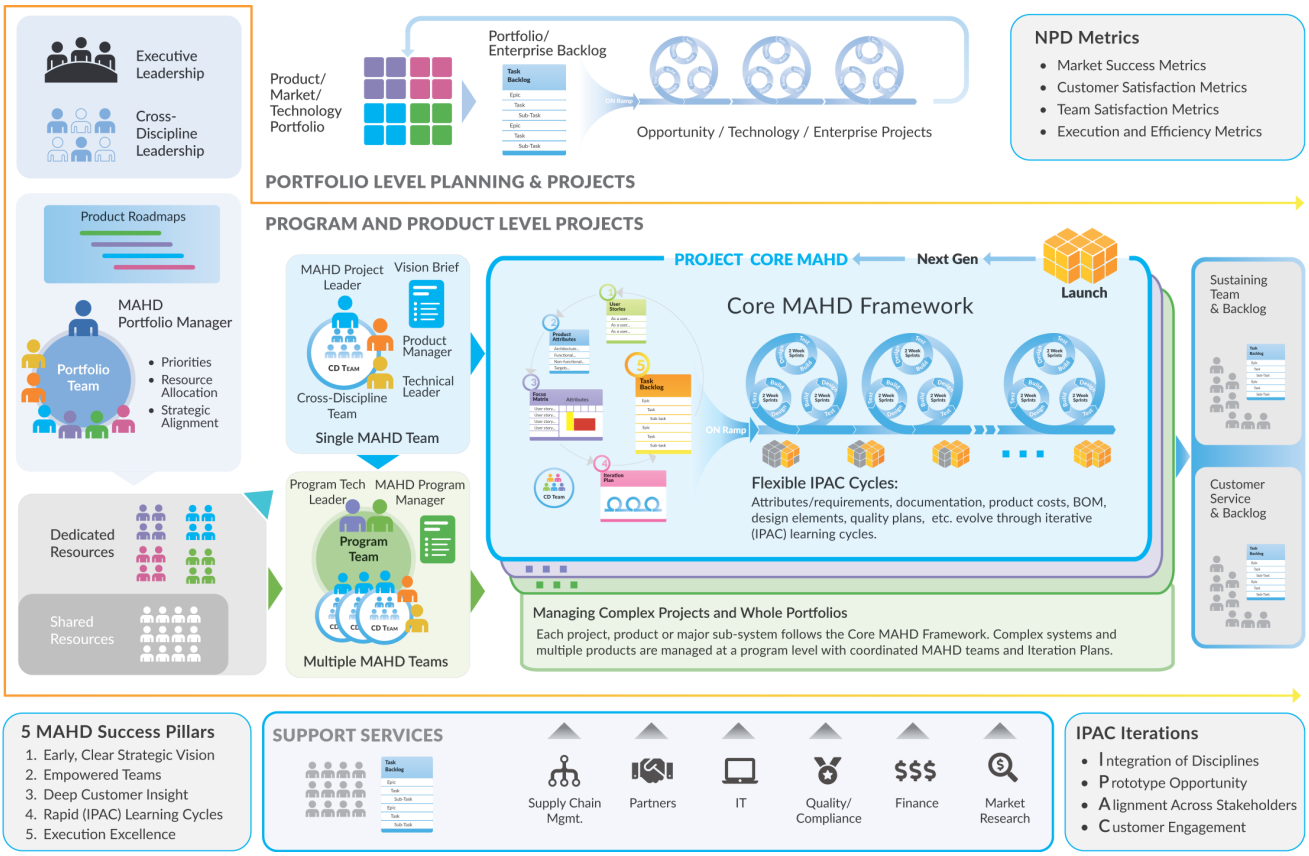


Figure 1: The Modified Agile for Hardware Development (MAHD) Framework™

Initiating an Electronics Agile Project – The MAHD On-Ramp

Once a project is selected, hardware development requires a different approach to initiation than software. While software teams can launch with a Minimal Viable Product (MVP) and add features over time, electronics products demand more upfront planning to ensure the right foundation.

Traditional methods often involve months of wasted effort, including building business cases, drafting Product Requirements Documents (PRDs), and negotiating features and schedules—much of which are based on assumptions that go unvalidated.

A better approach is to frame the problem and develop an optimal solution through rapid learning cycles. The MAHD Framework enables this by incorporating structured yet lightweight planning before execution begins. This includes:

- Design & architecture considerations
- Identifying dependencies & risks
- Defining prototype strategies & schedules
- Clarifying critical success factors

To streamline project initiation, the MAHD Framework introduces the Agile Vision Brief, followed by five cross-functional planning activities. Within days or weeks, teams:

- Align on strategic intent & customer outcomes
- Develop a high-level iteration plan
- Identify risks & innovation opportunities

Table 4 outlines the key activities and outcomes of the MAHD On-Ramp process. Each step moves the team toward identifying a path to project success and being ready to execute.

Table 4: MAHD On-Ramp Steps, Activities and Outcomes

MAHD ON-RAMP STEP	PURPOSE	ACTIVITY	OUTCOMES
Category	Define the project's strategic intent	Product management drafts a 1-2 page brief and aligns with stakeholders on the problem being solved	• Value proposition • Success factors • Project targets - Constraints
User Stories	Prioritize customer outcomes	Develop system-level user stories focused on defining electronic solutions	• Prioritized target customers • Prioritized customer needs • Prioritized customer outcomes
Product Attributes	Establish a preliminary solution framework	R&D responds to the Vision Brief & User Stories by identifying key solution attributes and technical considerations	• Defined solution elements • Subsystem structure
Focus Matrix	Identify risks, innovation areas, and key actions	The cross-functional team maps connections between User Stories and Product Attributes, then aligns on risks, innovation needs, and action plans	• Clear understanding of project elements • Identified risks & innovation areas • Prioritized action list • Preliminary prototype strategy & customer engagement plans
Initial IPAC Iteration Plan	Define an iterative learning & execution path	The team outlines the project scope, clarifies 1st IPAC Iteration goals, and identifies key milestones	• Clear project completion milestone • Defined 1st IPAC Iteration goals • Key milestones & prototype timing • Agreed velocity for success
Task Backlog Structure	Establish flexible, subsystem-aligned task management	Define how system deliverables, customer outcomes, and tasks will be tracked and structured	• Identified swim lanes • Initial tasks for 1st Iteration • Defined major solution epics

Tailored MAHD On-Ramps for Electronics Teams

While the MAHD On-Ramp steps apply to all electronics solutions, teams must consider where their target solution falls in the solution value chain. Table 5 briefly addresses how the focus of each MAHD On-Ramp activity may vary based on the difference between internally integrated solutions vs. sellable electronics.

Table 5: MAHD On-Ramp Considerations Based on Electronics Position in the Value Chain

MAHD ON-RAMP STEP	SELLABLE ELECTRONIC COMPONENTS OR DEVICES	ELECTRONICS AS A PRODUCT OR SYSTEM ENABLER
Vision Brief	Focused on product success factors and constraints	Focused on system success factors and constraints
Customer Outcomes	**Primary:** Target market needs	**Primary:** Internal customer needs **Secondary:** Marketable solution needs
Product Attributes	**Flexible:** Solution optimized per market needs	**Constrained:** Limited by system limitations and interfaces
Focus Matrix	Finding connections between external customer needs and solution attributes	**Two levels:** Connections between internal and external customer needs and solution attributes
Iteration Plan	**Independent:** Focused on externally oriented solution milestones **Prototypes:** Independent, aligned with optimal external integration validation	**Dependent:** Focused on internally oriented, aligned solution milestones **Prototypes:** System planning to identify optimal functionality testing
Backlog Structure	Dependencies tracked primarily across subsystems and disciplines	Dependencies tracked across subsystems and disciplines as well as within the broader system solution

Rapid Learning and Execution Cycles – IPAC Iterations

All agile development practices rely on iterative learning and execution cycles. In Scrum-based software development, these cycles, or sprints, typically last 1 to 4 weeks, resulting in functional, demonstrable software. For hardware, the MAHD Framework adapts this approach by introducing two levels of iterations, with IPAC Iterations at the system or product level.

How IPAC Iterations Work

Each IPAC Iteration spans one to six sprints and focuses on:

- System-level integration across disciplines and subsystems
- Steady development of all product and system deliverables
- Validating the overall solution
- Reducing technical and commercial risk

A common outcome is a demonstrable output (a form of prototype discussed below) that helps resolve key uncertainties.

Why IPAC Iterations Matter for Hardware

Unlike software, hardware development requires coordination across mechanical, electrical, and firmware teams. IPAC Iterations help by:

- Aligning subsystems and disciplines to key milestones
- Providing a structured schedule to track and communicate progress
- Reducing overhead, allowing teams to manage execution-level sprints efficiently

The Four Dimensions of IPAC

To ensure each iteration drives meaningful progress, the concept of IPAC goals was added to MAHD 2.0 to help teams define goals across four key dimensions:

I - Integration: What functionality should be integrated across disciplines or subsystems

P - Prototype: What prototype can reduce risk and provide valuable insights?

A - Alignment: What external dependencies (suppliers, partners, stakeholders) need to be addressed?

C - Customer: How can customer engagement validate decisions and improve outcomes?

By systematically addressing these four directions, teams accelerate learning and execution, ensuring projects progress at the right velocity to meet their goals.

Tailored MAHD Execution for Electronics Teams

Like the on-ramp considerations, the MAHD Framework execution steps apply to all electronics solutions. However, many of the considerations shown in Table 5 will continue throughout the project to ensure the team is focused and aligned correctly. Table 6 briefly addresses how the focus of MAHD execution activities may vary based on the difference between internally integrated solutions vs. sellable electronics.

Table 6: MAHD Execution Considerations Based on the Electronics Position in the Value Chain

MAHD EXECUTION STEP	SELLABLE ELECTRONIC COMPONENTS OR DEVICES	ELECTRONICS AS A PRODUCT OR SYSTEM ENABLER
IPAC Iteration Planning	Continued focus on the market success factors, areas of innovation, and gaining external customer feedback	Continued focus on system success factors and constraints, often leveraging a team-of-teams approach to ensure system-level planning and alignment
Sprint Planning	Subsystems and/or discipline teams conduct sprint planning while staying focused on IPAC goals and milestones	Enhanced collaboration is often needed with joint sprint planning to align internal system dependencies
Other Considerations and Project Needs	• Keen awareness of external market factors • System integration needs of external solutions and customer feedback loops • Strong product manager/owner alignment	• Keen awareness of internal system challenges • System integration and broader needs of the external solution • Strong, agile project leadership and collaboration

Making Progress: Iteration→Sprint… Sprint→Iteration: Repeat

Once IPAC Iteration Planning is complete and teams commit to milestones and goals, swim lane teams update the backlog and prepare for sprint planning. In software agile, sprints drive progress, but the day-to-day execution of agile for software practices leaves most hardware teams frustrated. For example, whole team task estimation and daily standups are core agile practices, but hardware teams find them misguided and often unnecessary.

The MAHD Framework establishes defined practices for IPAC Iteration planning and reviews. The framework adapts more detailed execution practices to maximize value and minimize overhead while still maintaining agile principles.

Key Adaptations in Agile Execution for Hardware

1. Sprint Planning

   To balance efficiency and agility, the MAHD Framework recommends:

   - Smaller, subsystem or discipline-focused teams plan two-week sprints
   - Cross-team collaboration happens as needed to meet IPAC Iteration goals
   - Teams maintain autonomy in determining the most efficient way to execute

2. Standup Meetings

   Unlike software, daily standups may not be practical for hardware teams. The MAHD Framework recommends:

   - Starting with bi-weekly standups, adjusting as needed
   - Keeping meetings concise and focused on blockers and key collaboration needs

3. Task Estimation

   Traditional Scrum methods use Fibonacci-based or similar task estimation, where all team members contribute. However, in hardware:

   - Subject Matter Experts (SMEs) are often the only ones qualified to estimate and execute specific tasks
   - MAHD teams use point-based estimation, but only SME's contribute and make the final commitment

4. Acceptance Criteria

   Defining «done» in hardware is more complex than in software. For example:

   - In software, a user story like «As a user, I want to log in» has clear acceptance criteria
   - In hardware, a task like «Develop a preliminary PCB layout» might require system-level testing several iterations later

   The MAHD Framework shifts acceptance criteria from task-level to IPAC Iteration goals, ensuring system functionality, customer feedback, or demonstrable output are clearly defined for each iteration.

Regardless of how the hardware teams choose to plan and execute sprints, throughout each IPAC Iteration, teams

- **Align across swim lanes** to integrate components and meet IPAC goals.
- **Deliver demonstrable outputs**, often through prototypes.
- **Engage with customers** to gather valuable feedback when possible and valuable.
- **Conduct retrospectives** to refine processes and improve efficiency.

As IPAC Iterations advance, teams make collaborative trade-offs, management gains clear visibility, and tangible progress is made. This approach accelerates development by 25% to 50%, leading to an optimized, manufacturing-ready solution—far outperforming traditional waterfall methods.

Key Concepts for Electronics Agile Success

Agile for electronics adoption starts with the right mindset and making the critical agile modifications for hardware, as discussed. Executing the MAHD On-Ramp to initiate projects and implementing IPAC Iterations for planning and alignment are foundational to leveraging agile principles. However, there are several key concepts that, when embraced, almost ensure you get the desired benefits.

Rethink "Requirements"

In waterfall processes, defining and scrutinizing requirements is central to project initiation. Without them, R&D teams feel stuck. Yet even the most detailed requirements documents are full of assumptions—about customer needs, the best solution, and feature priorities. They often blend wish lists, overly specific details, and vague directives like "easy to use."

This rigidity leads to prolonged debates between R&D and business stakeholders, wasting time, energy, and political capital as teams struggle to separate aspirations from actual commitments. A better approach is to skip rigid requirements entirely. Instead:

- Start with clear strategic intent and defined customer outcomes.
- Collaborate as a team to refine the solution through rapid learning cycles.

For electronics projects, this may seem challenging due to their reliance on technical specifications. But once teams move beyond rigid requirements, they gain the flexibility to solve real problems, maximize customer value, and deliver better results.

Write System Level User Stories

Without traditional «requirements,» teams must define and prioritize customer outcomes in a clear, actionable way. In software, teams use user stories to capture needs—an approach that works well since software is directly experienced by users.

The MAHD Framework also uses user stories but at the system level rather than focusing on individual features. Customers experience products holistically, not as isolated features.

For example, an electronics solution may include chassis materials, I/O ports, and LEDs, all working together to meet customer needs. Instead of documenting each feature separately, the focus is on intended solution outcomes, allowing teams to prioritize features and specifications as they evolve to maximize customer value.

Embrace Matrix Thinking

Traditional requirements often mix assumed customer needs with predefined solutions. For exampl

"The unit shall have a copper RF shield with 20dB attenuation between 100 MHz – 5GHz."

This statement assumes that:

- RF shielding is necessary for the desired outcome.
- Copper is the best material choice.
- 20dB attenuation is optimal.

But why is RF shielding important? How was 20dB determined? Could a different approach provide better value? By framing this as a fixed requirement, teams risk over-engineering, missing alternative solutions, or failing to validate the real need.

A Smarter Approach: Separating Outcomes From Product Attributes

The MAHD Framework addresses this by distinguishing User Stories (customer outcomes) from Product Attributes (technical capabilities). Teams use a structured tool called the Focus Matrix, which helps:

- Identify how multiple User Stories connect to specific Product Attributes
- Align customer expectations with engineering decisions
- Prioritize development based on impact and feasibility

Unlike software Agile, where features are prioritized independently, electronics development requires holistic planning, considering dependencies like power management, signal integrity, and manufacturability.

Applying the Focus Matrix in Practice

Instead of locking into a fixed requirement, a better user story might be:

"As a user, I need reliable wireless connectivity so my device maintains a strong signal in varied environments."

This leads to multiple Product Attributes, such as:

- **Antenna design** – Optimizing signal reception
- **Power efficiency** – Reducing battery drain during transmission
- **RF shielding** – Minimizing external interference

By mapping these attributes to the User Story, teams can:

- Prioritize engineering efforts.
- Refine specifications iteratively.
- Ensure the final product efficiently meets customer needs.

This approach also helps teams consider multiple customer layers in the value chain, ensuring that every decision aligns with real market needs rather than rigid assumptions.

Align with Your Software Teams' Agile Approach

As teams execute IPAC Iterations, aligning subsystems—especially between electronics and software—is crucial to managing dependencies and validating functionality early. The MAHD Framework simplifies this through joint Iteration planning, ensuring seamless collaboration regardless of whether software teams use Scrum, SAFe, or another approach.

By establishing IPAC milestones and goals, each subsystem operates in shorter sprints, coordinating along the way to achieve aligned objectives. Joint planning and synchronized processes are essential for electronic devices that integrate into larger system development efforts.

Always Be Ready for Executive Reviews (And Gates)

Many teams must align with well-established Stage-Gate or phased NPD systems. With each IPAC Iteration, the MAHD Framework provides:

- Visibility into overall project status
- Clear risk assessment and tradeoff decisions
- Progress on downstream deliverables
- Demonstrable results

While traditional gate reviews require weeks of preparation, MAHD teams are always ready for executive reviews with minimal effort. Since status updates are frequent, formal gate meetings often become unnecessary.

Apply Strategic Prototyping Strategies

One final key concept is the importance of flexible prototyping.

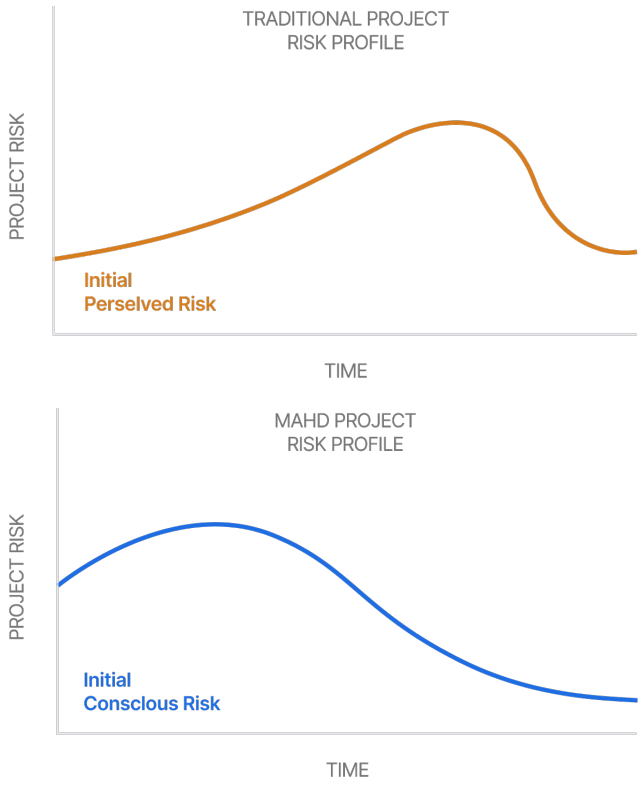
Agile emphasizes frequent prototyping to validate both technical feasibility and market fit. However, software-driven agile coaches often suggest hardware teams build a functional prototype every sprint, which is often impractical, costly, and unnecessary. A better approach is to tailor prototyping to project complexity and risk:

- Simple projects with minimal unknowns may need few or no prototypes.
- Complex projects benefit from frequent, targeted prototypes to reduce risk.
- Teams can mitigate risk through parallel learning efforts, breaking solutions into «vertical slices» and using partial physical prototypes or simulations for technical validation and mockups, digital models, or even preliminary brochures for early customer feedback.

Developing a flexible, risk-based prototype strategy is a core outcome of the MAHD On-Ramp, ensuring teams balance speed, cost, and learning efficiency.

The Bottom Line – A Better Risk Profile

When combining agile principles, modified tactics, and several key concepts, the MAHD Framework delivers a key advantage over waterfall approaches—reduced risk. In waterfall, risk accumulates as integration, testing, and key decisions are delayed until late in development. The MAHD approach, by contrast, tackles risks from the start, ensuring more predictable outcomes regardless of project objectives. By lowering risk, you can go faster, deliver higher value, and obtain more predictable outcomes.



How Altium Supports Agile Hardware Development

While the MAHD Framework can provide the structure, methods, and way of working for electronics teams to get the benefits of agile, teams can increase efficiency and collaboration with targeted solutions Altium offers. As agile allows the team to flexibly define the solution as the team learns, it's imperative to leverage tools that enable tracking and maintaining a source of truth as the solution evolves.

Altium supports agile electronics development, combining all aspects of electronic design and development, helping organizations deliver better products faster than ever. By breaking down data silos and enabling real-time collaboration, Altium solutions ensure that hardware development teams can fully embrace and benefit from agile methodologies.

Digital Evolution Management

Agile development thrives on adaptability, requiring teams to manage product evolution digitally. Altium solutions ensure that every modification aligns with the project's evolving goals.

- **Track Design Changes and Decisions Across Disciplines:** Ensure smooth collaboration and coherence in multi-disciplinary product development.
- **Ensure Traceability with Version Control:** Keep track of every design decision, gaining a clear audit trail from concept to completion.
- **Eliminate Manual Updates:** Reduce the overhead associated with traditional document management, focusing instead on live, actionable data.
- **Synchronize with Jira:** If you're using Jira in your hardware development process to track iteration goals, sprint objectives, and task backlogs, you can sync it with your ECAD tool—thanks to Jira Integration. Electronics teams can automatically synchronize tasks, issues, statuses, and comments between Altium and Jira, enabling seamless collaboration, real-time cross-discipline tracking, and enhanced project visibility.

Rapid Prototyping

The essence of agile in hardware lies in the ability to prototype swiftly. Altium solutions support this need through its comprehensive capabilities, efficiently facilitating the transition from digital models to physical prototypes.

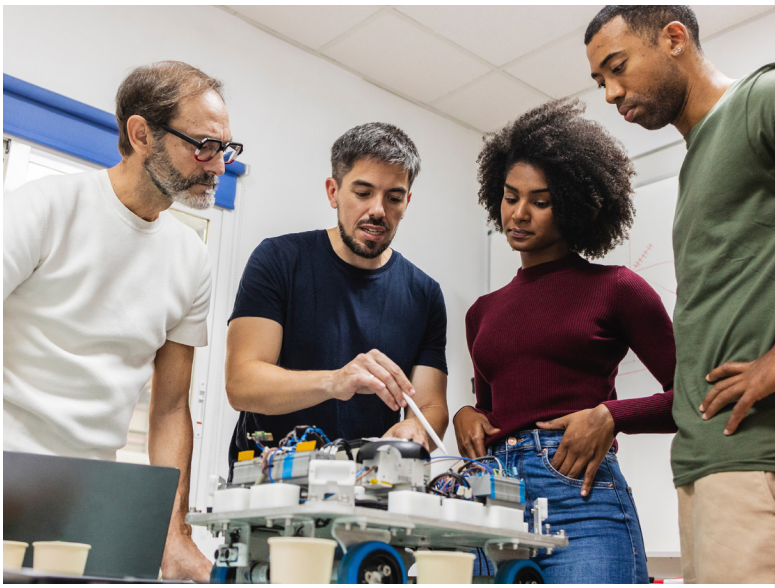
- Integrate with MCAD tools to swiftly build digital prototypes.
- Bridge the gap between engineering and procurement to reduce delays and ensure prototyping resources are available when needed. Order prototype components directly from the design environment.
- Accelerate prototype assembly with an intuitive UI combining the latest design data and BOM. Quickly identify component locations for accurate placement and faster build times.

Customer Feedback Loops

Agile thrives on customer interaction and feedback. While Altium solutions may not collect feedback directly, they excel in incorporating it into actionable design changes.

- **Trace Requirements to Attributes:** Ensure customer needs and feedback directly influence design specifications and attributes.
- **Stay Aligned with Centralized Requirements:** Ensure your team always has access to the latest requirements with a centralized, cloud-based requirements management. No more errors caused by scattered spreadsheets and disconnected documents. Link requirements to designs, test cases, and documentation, enabling full traceability and compliance throughout the development lifecycle.
- **Customize Workflows:** Adapt workflows to incorporate feedback at any stage of the development process, ensuring products evolve in response to real user needs.

Altium solutions provide tracking, traceability, and the transparency necessary to enable agile success.



Additional Resources

About the MAHD Framework

The Modified Agile for Hardware Development (MAHD) Framework™, introduced in 2017, helps hardware teams apply agile principles effectively. Since then, it has enabled teams across various industries to accelerate development by 25% to 50%, while reducing risk and maximizing customer value.

MAHD's success comes from adapting agile tactics and introducing hardware-specific methods. It has proven ideal for managing complex products and systems that integrate mechanical, electronic, and software components.

Learn More About the MAHD Framework

Contact the MAHD Team for Expert Guidance on Adapting Agile to Hardware

About Altium

Altium empowers secure, seamless, and efficient collaboration across the electronic product development lifecycle. By connecting all stakeholders, from concept to production, Altium transforms the way teams design, develop, and deliver electronic systems. Its solutions unify the critical aspects of electronics design within a single, integrated environment, fostering greater visibility, alignment, and innovation at every stage.

Learn More About Altium Solutions