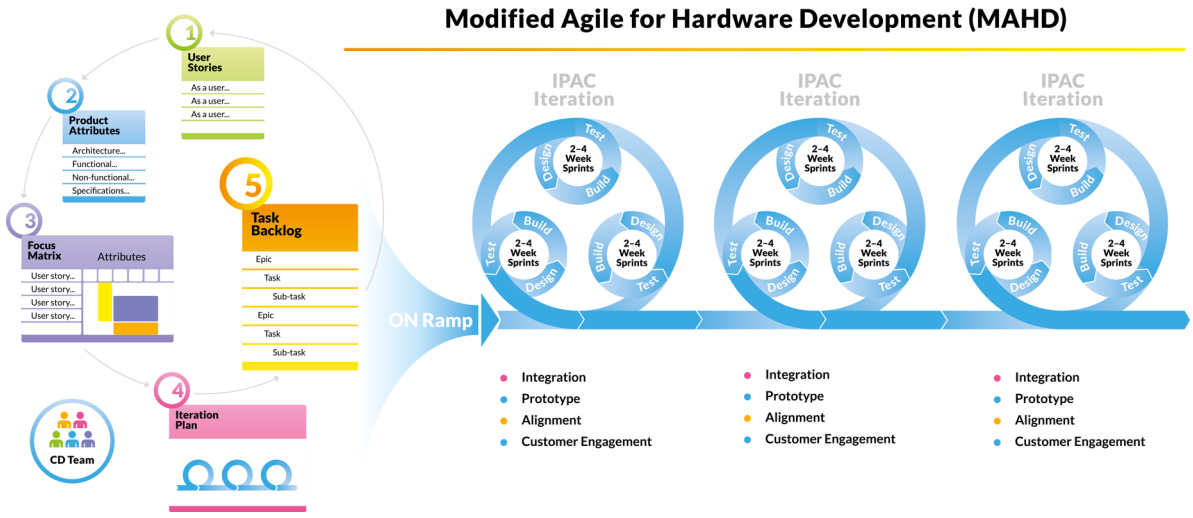
WWW.AGILEFORHARDWARE.ORG



An Introduction to

MAHD

Modified Agile for Hardware Development

Applying Agile Principles to Physical Products When Making Changes Is Difficult

By Dorian Simpson and Gary Hinkle

CONTENTS

What's Inside

THE GOALS FOR THIS E-BOOK:

1. *Provide an introduction to the Modified Agile for Hardware Development Framework.*
2. *Explain the key differences between hardware and software Agile methods and why a modified approach is both desired and necessary.*
3. *Explore the major elements of the MAHD Framework and what makes it unique from Scrum or other SW-based Agile methods.*
4. *Share tips on how to get started with the MAHD Framework.*

Is Agile Right for Hardware?

INTRODUCTION

Agile methods have taken over the software industry. Developers and leaders have discovered that traditional waterfall processes don't work because the upfront unknowns are too significant to accurately write requirements and estimate work. So a new way — Agile — was created and embraced. All good. But what about products that have mechanical and electronic components? Can products that range from trash cans to complex medical devices benefit from Agile's benefits? Yes. The principles are sound, but Agile was not designed to work for hardware, just as traditional waterfall processes were not designed for SW. Agile methods require modification to support the needs of hardware products where making changes are costly, partial products are difficult to test with real customers and definitions must be frozen for production. This led the need to develop the Modified Agile for Hardware Development (MAHD) Framework — an Agile initiative to gain the benefits of Agile while recognizing hardware's unique needs.

Before moving on, consider the following questions and answer for yourself, "Is Agile right for your hardware development efforts?"

1. *Is it **difficult to get clear product requirements** before development starts?*
2. *Does risk accumulate throughout the development cycle as milestones are missed and **changes force rework**?*
3. *In order to get accurate and valid feedback from customers, do customers need to first **experience** your product?*
4. *Is **market success based on innovation** in several focused product areas or key specifications?*
5. *Does management need clear guidance from the team on where the primary **project risks** are and the plan to mitigate these risks?*
6. *Do individuals or teams **wait long periods** before their work is validated or integrated into the product?*

If you were able to answer "yes" to most of these, then the Modified Agile for Hardware Development Framework may be a good fit for your organization.

In the following sections, we'll address each element of the MAHD Framework, why it's different than SW-based Agile methods and how you can get started applying Agile principles to your development efforts.

The MAHD Framework - An Introduction

SIMILAR TO AGILE FOR SW METHODS, BUT WITH CRITICAL DIFFERENCES

The MAHD Framework uses the principles of Agile to develop physical products in less time, with reduced risk and with higher customer satisfaction. Many companies have attempted applying Agile for SW methods directly to physical products with mixed results. Teams often struggle since current Agile steps, techniques and even language were not optimized for hardware development. A modified Agile approach can leverage the power of Agile, while addressing the unique needs of hardware development.

As shown in the framework on the following page, MAHD has many elements of Agile that may be familiar to you. Developers start with user stories, a backlog is kept to prioritize tasks, and it includes iterative development cycles. But there are also key differences. A summary of these include:

The On-ramp
One of the basic principles of Agile is to develop very little formal documentation to keep the team focused on the most valuable activities. This is true of Agile for hardware methods also, but the nature of physical products requires additional upfront planning steps and often more documentation.

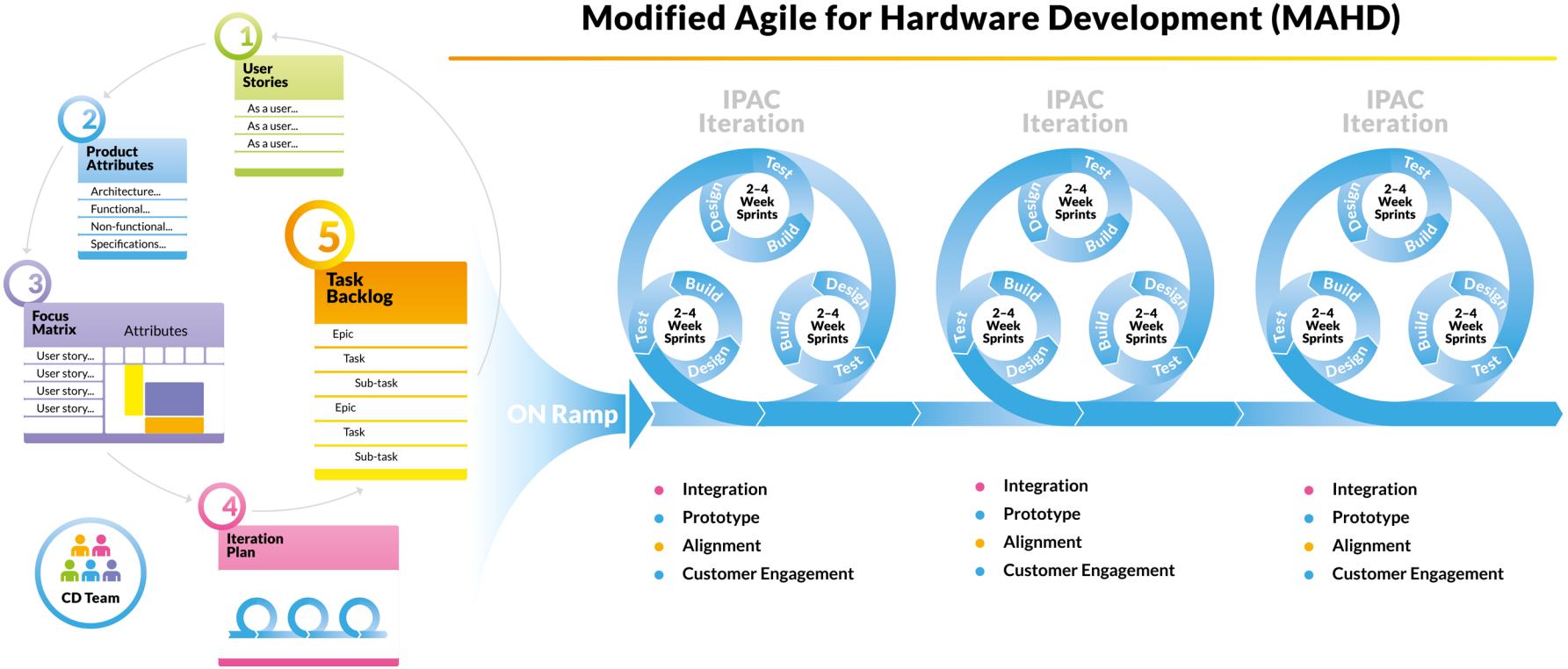
Iterations and Sprints
Just as with Agile for SW methods, the MAHD Framework has at its core the concept of short development and learning cycles. But as the model below shows, the MAHD Framework includes two levels of cycles that must be considered to accommodate the needs of hardware development.

Teams and Deciders
Another foundation of Agile is the use of small autonomous teams led by product owners and "Scrum masters" with as little governance as possible. Again, this is similar for hardware, but the teams, project leaders and deciders often have different roles, titles and focus.

In the following sections, we describe the core of the MAHD Framework focusing on a single project. As you extend MAHD to more products and larger systems, MAHD can also scale to any level of complexity as summarized in the final section. To explore an interactive model of the Complete MAHD Framework, visit: www.agileforhardware.org

For those familiar with Agile for SW methods, page 16 has a comparison of the MAHD Framework versus Scrum, the most commonly used Agile for SW methodology.

MAHD FRAMEWORK



The Modified Agile for Hardware Development Framework uses the principles of Agile for SW, but has significant differences to support the unique needs of physical product development.

MAHD ROLES: It's Agile, But Roles Need Modification Too

Before diving into the elements of the MAHD Framework, let's look at some of the roles defined in Agile methodology and how these roles might be different for HW projects. Rather than focus on the wide range of titles, we will focus on four types of roles - deciders, leaders, doers and facilitators.

As the diagram shows, the roles are similar, but there are a couple of important distinctions. The first is the "decider" role. For software, deciders are typically defined as a Product Owner. This person writes user stories, prioritizes the backlog and approves all sprint tasks. They represent the customer in all aspects of sofware development. For HW this role will often be filled by a Product Manager. This person may or may not work as closely with execution teams as SW Product Owners. They must understand and consider the whole product with all of the integrated components and make decisions that may have cross-functional implications.

Another clear difference in the MAHD Framework is the "facilitator" role. Software often defines a Scrum Master that has deep expertise in software development. For HW, we need someone who can manage across disciplines. This is often a project management role, but MAHD defines an Agile Project Manager as one who must also be skilled in the Agile process to guide everything from backlog development to successful iteration planning. The specific titles are not as important as determining who will lead teams, who will make decisions and who will facilitate the process.

Deciders

Those who define the product and make tough priority decisions.

SW: **The Product Owner**

HW: **The Product Manager**

Leaders

Those who lead and manage functional teams.

SW: **Development Leads**

HW: **Functional Leads**

Doers

Those who do the work.

SW: **Developers, testers, analysts, designers, etc.**

HW: **Developers, engineers, testers, designers, etc.**

Facilitators

Those who drive the process.

SW: **Scrum Master**

HW: **Agile Project Manager/ MAHD Master**

Get complete role descriptions by joining our free community at: WWW.AGILEFORHWARE.ORG

Kicking Off a MAHD Project

One of the goals of Agile methods is to enable faster project kickoffs. With traditional methods it can take months to build a business case to gain project support, more time to develop a "complete" Product Requirements Document (PRD) and then more time for marketing and R&D to negotiate features and develop detailed schedules. Most of this time is wasted on things the company already knew, assumptions that never get tested and schedules no one believes.

To kick off a MAHD project, a better approach is to use a light product-market description document that provides concise information the development team can use to get started. The "Agile Product Brief" shown below summarizes the market situation, clarifies the customer and their needs, establishes project goals and identifies the high level value drivers that lead to purchase in the market. The example shown here is from our Step-by-step guide to MAHD which follows the JavaBrew team as they use Agile methods to develop a new, innovative coffee maker.

Transforming to Agile methods often does require a change to a company's culture including getting more comfortable with uncertainty and having trust that the team will evolve the product in the right direction as they learn. In the next section, we'll look at each step of the MAHD On-ramp which provides a set of five collaborative activities designed to:

1. Quickly get marketing and R&D teams on the same page
2. Identify the aspects of the project that will lead to success and/or create risk
3. Develop a high-level plan to begin execution

Agile Project Brief

Market Overview

- Voice-enabled devices are growing. After 2 years are already in 24% of US homes
- Amazon Echo dominates the market with 75% penetration
- Smart coffee maker category growing 22% YOY
- Many makers are adding "smart" features – only one is voice-enabled today

Target Customer

- US home consumer (to start)
- Primary target: Male, 25 to 40 years of age, household income >$150K/year
- Tech savvy: Premium smart phone owners, smart device users
- Coffee lovers: Drink 2-5 cups of coffee/day. High quality. Personal bean preference.

Value drivers (reason to buy):

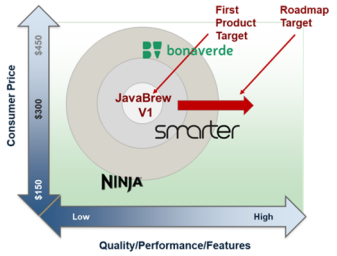
- Attractive design: Pleasing, fit with decor, clean, modern
- Quality of coffee: Taste, consistency, flexibility
- Long term experience: Maintainable, functional, durable
- Smart: Easy, cool, intuitive, new use cases

Launch Goals:

- Retail Price: $299
- Wholesale price: $170
- Target cost: $100
- JavaBrew Margin: ~40%
- Target launch: July 31, 2020
- 2020 Unit target: 15,000 units

Product-market Positioning

- Initial JavaBrew product to enter smart market, roadmap adds product SKUs
- High value, premium product based on intelligent functionality and design not just being "smart"
- "JavaBrew's Smart Coffee Series delivers amazing coffee exactly how you want with an intelligent and intuitive design"



To learn more about this tool, download our 9-part Step-by-Step Guide: WWW.AGILEFORHWARE.ORG/STEP-BY-STEP-MAHD

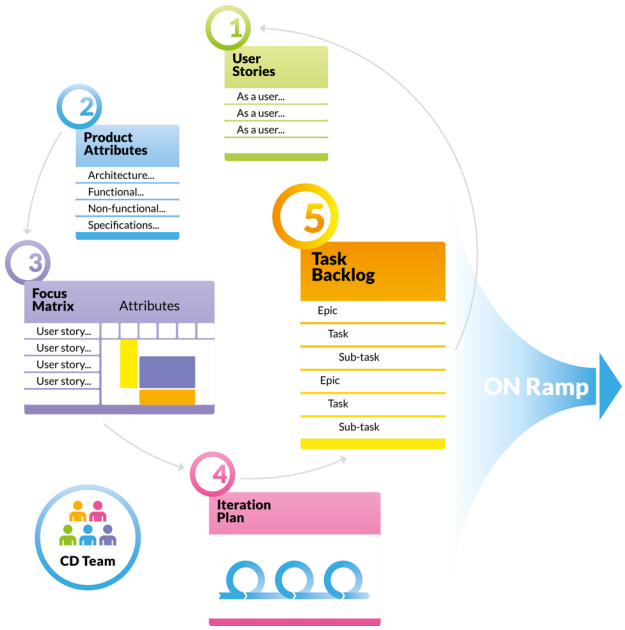
Preparing for Agile Success with the On-ramp

IT MAY LOOK "UN-AGILE" BUT HARDWARE REQUIRES DEEPER UPFRONT PLANNING

One of the major challenges that the MAHD Framework addresses for hardware is the need to think about the whole project before getting started with execution. Designs, architecture, dependencies, high-level iterations and even the schedule need to be considered before diving into the work itself.

As we'll describe, the MAHD On-ramp may take a few days or up to a couple of weeks, but the results are well worth the effort in setting the team up for success. While many of the elements are similar to those found in Agile for SW, each element must be modified and we'll introduce one new element to the MAHD Framework to support hardware development efforts. The following pages will look at each of these in more depth.

ELEMENTS OF THE MAHD ON-RAMP



1. **User Stories** - Hardware companies attempting to apply Agile struggle with user stories. MAHD addresses this by taking a system approach to user stories when defining physical products.
2. **Product Attributes** - While Agile for software does away with requirements (which are replaced with user stories), product attributes still have a necessary purpose in Agile for hardware.
3. **Focus Matrix** - This is a new element to Agile introduced in the MAHD Framework and builds a necessary bridge to drive iteration planning, Agile thinking, innovation and development focus.
4. **Iteration Plans** - Similar to a SW release plan, but quite different. The MAHD Framework uses two levels of development cycles. MAHD Iteration Planning and IPAC Cycles create the game plan for success while sprints provide the execution details.
5. **Task Backlog** - Notice that this is not the product or feature backlog. It's different, but related and focuses on specific team tasks that must be accomplished each sprint.

While these activities may look like early stages of a waterfall-oriented process, this doesn't mean the team writes a detailed product requirements document, a complex Gantt chart or gets sign-off by 12 managers. It does mean that the team has thought through the project details in enough depth to confidently get started with Agile development.

Rethinking User Stories for MAHD

HARDWARE FOLKS STRUGGLE WITH USER STORIES... AND FOR GOOD REASONS

User stories are a critical starting point for Agile for SW methods since they provide the backlog items, are groomed directly into features and sprint tasks and form the basis of the product definition. In essence, user stories ARE the product requirements for software. For hardware we need to rethink this. For example:

User stories go something like this...

> ***"As a user, I want to be to able to quickly log in so that I can access my account."***

Software developers know what to design and almost how to implement it.

Now let's try it for hardware. Let's assume you're planning to develop a new fork lift and you write this user story:

> *"As a user I want to be able to quickly pick up my material so that I can save time moving inventory."*

Does a developer of hardware know what to do? Probably not. There are too many facets of the problem to solve. The implementation might involve the speed of the fork lift, the accuracy of the fork attachment, the orientation of the inventory and many other factors. Rather than specific features or tasks, these user stories for hardware become customer goals, rather than product requirements.

Prioritized User Stories

USER STORIES ARE STILL VALUABLE, BUT INSUFFICIENT

The previous example might lead you to think that user stories are not appropriate for physical products, but this would take us too far back to traditional waterfall methods where customer needs often take a back seat to the desire to develop detailed product requirements.

User stories have their place in the MAHD Framework and are necessary to create a focus on the needs and priorities of customers as well as to clarify results customers are trying to achieve. However, since user stories for physical products cannot typically be directly translated into features, functions or tasks, they become the starting point for developing a task backlog rather than backlog items themselves. Once you have written MAHD User Stories, it takes several more steps to identify the specific backlog tasks.

Next, we'll need to consider some of the hardware-centric steps that you won't find in Scrum or other SW-based Agile processes — product attributes and the focus matrix.

Product Attributes: Describe It, But No PRDs

WRITING PRODUCT REQUIREMENTS IN AN AGILE WORLD

Just the term "requirements" is almost anathema to Agile purists. They know that customers typically don't have requirements, they only have needs and goals to accomplish that can be described as user stories. But hardware is different. Physical products often have limitations in components, must meet target specifications or must accommodate pre-existing interfaces. Certainly we could describe the details through user stories, but this approach is tedious to document and obscures the purpose of user stories.

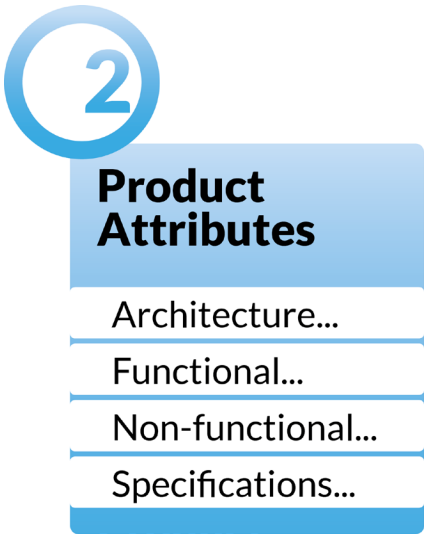
For example, if we consider our forklift user story again,

> *"As a user I want to be able to quickly pick up my material so that I can save time moving inventory."*

We could enumerate the specifics of the features and functions to satisfy this user story with more user stories, such as,

> *"As a user, I want the fork lift to be able to lift materials at a rate of three meters per second."*

This is not really a user story, but more of a target specification or perhaps part of the acceptance criteria of the user story.



Product Attributes

To be clear, we are not advocating that MAHD Framework practitioners write detailed product requirements documents that are associated with waterfall processes. However, concisely describing the product vision and rough architecture as well as listing the anticipated functional and non-functional product attributes is important as we'll see in the next step. Keep in mind that "attributes" as defined in the on-ramp aren't rigid specifications, but really just a high level summary of what might be expected to be in the product to act as a starting point to seed the Agile process and guide it in the right direction. Each attribute will get refined into specific features, functions and requirements that satisfy customer needs (the user stories) as the Agile process moves forward.

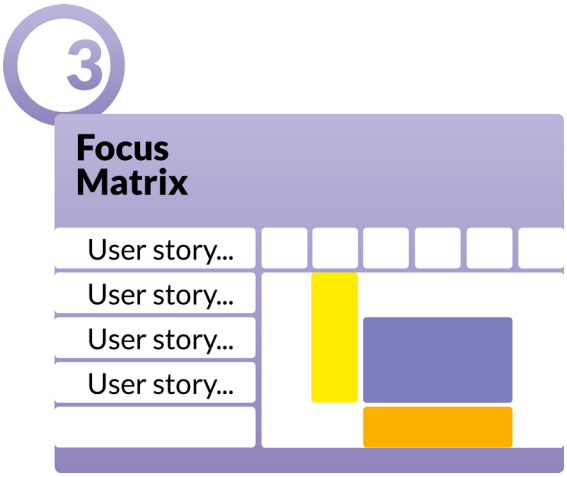
To summarize, without describing the product attributes at some level during upfront planning it will be difficult to continue with the on-ramp and develop the backlog of tasks as well as to identify areas of functionality that will be used for iteration planning, prototyping and the focus of innovation efforts.

Driving Agile Priorities with a Focus Matrix

FINDING IMPORTANT CONNECTIONS BETWEEN USER STORIES AND ATTRIBUTES

The Focus Matrix is a MAHD element you won't find in any version of Agile for SW methods since it is not necessary for SW development. However, matrix thinking has been a powerful tool for hardware development for decades and is valuable for Agile for hardware planning for one good reason — user stories will be satisfied by a range of product attributes, and various product attributes will contribute toward satisfying a range of user stories.

This n-to-n relationship cannot be documented, understood or analyzed without some form of relational matrix. For those who have studied Total Quality Management (TQM), you may be aware of the House of Quality. This matrix planning tool is used to define the relationship between customer desires (user stories in Agile terms) and product capabilities (features). While the TQM House of Quality is far too complex for Agile purposes, the rationale behind it is critically important and can help MAHD practitioners focus on the most important tasks.



Focus Matrix

Considering the diagram on the right, the MAHD Focus Matrix has two dimensions. The left hand side of the matrix shows prioritized user stories. These are customer needs and goals. On the top of the matrix are the range of prioritized functional attributes from our first two steps of the on-ramp. The relationships between these factors become the basis for iteration planning, prototype plans and dependency identification.

Consider our fork lift example. The user story, ".... *quickly pick up my cargo..."* would appear on the left and could be satisfied by a range of attributes that will be organized on the top, such as: The speed of the lift, the lift attachments, the accuracy of steering, optical recognition to determine the orientation of the inventory, etc. The team would then determine which attributes contribute to satisfying the highest priority user stories and develop a plan of execution for how to prototype the attribute, what questions need to be answered and how to get customer feedback.

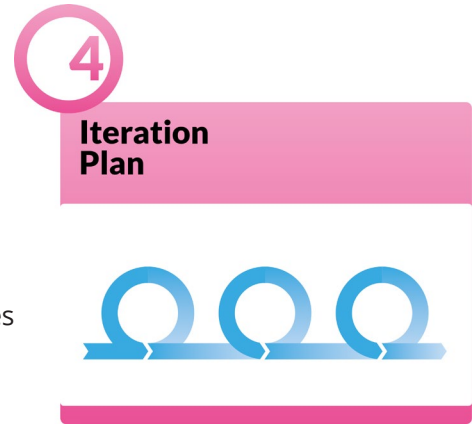
Successfully working through the Focus Matrix leads to the next MAHD On-ramp activity, iteration planning. The result becomes the high-level project plan that is necessary for managing risk, schedules, resources and critical IPAC milestones.

Iteration Planning in a MAHD World

CONNECTING USER STORIES, REQUIREMENTS AND ITERATIONS

As discussed earlier, at the core of any Agile methodology are iterative development and learning cycles. For Scrum-based Agile for SW, "sprints" last from 1-to-4 weeks and result in working software that can be demonstrated to users. The MAHD Framework modifies this concept and considers two levels of iterations. At the high level are IPAC Iterations (or Whole Product Iterations). These iterations lead to larger deliverables where components of the solution are integrated and a key result is often a demonstrable prototype that can be viewed and validated with internal and/or external stakeholders (i.e. customers).

The MAHD Framework also uses shorter execution cycles within IPAC iterations. These are 2-to-4 week MAHD sprints. The distinction is important for hardware-oriented products since it is unlikely you'll be able to create a demonstrable product with every sprint that can be put in front of customers, but working toward prototypes at the higher level iterations is critical to being Agile. Without customer interaction at key phases of product development, you will, by default, be reverting back to waterfall practices.



Iteration Plan

To develop an iteration plan, consider four types of milestones for each IPAC Iteration:

1. **I - Integration** - What is the right timing and functionality that should be integrated from each discipline (electronics, software, mechanical, etc.) as well as what technical questions need to be answered?
2. **P - Prototype** - How will you develop a prototype plan at different levels of sophistication in order to gain real customer insight? (Prototypes can range from drawings to nearly full functionality.)
3. **A - Alignment** - What are the major dependencies or stakeholders you need to get aligned at each point?
4. **C - Customer** - What key questions do you need to answer to validate customer acceptance and what is your feedback plan to answer them?

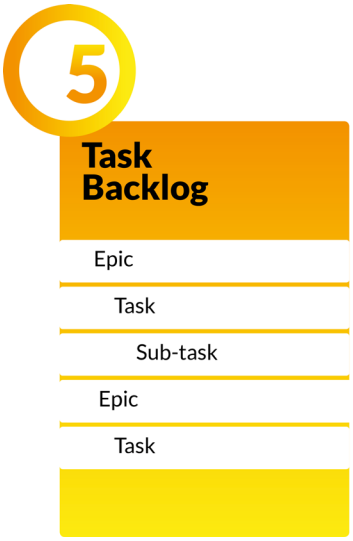
Back to our fork lift example. Let's assume you decide that determining the orientation of inventory is crucial to satisfying a high priority user story. You then plan to focus on developing a solution and early prototype in an early iteration in order to validate the technology and customer acceptance. As you plan for iterations, this level of prototype might not be possible until the second iteration, so you decide the first iteration deliverable will include an animated video explaining and demonstrating the solution for customers to get early feedback.

At this point of the MAHD On-ramp, you're not planning detailed tasks or sprints, but only identifying important IPAC milestones, key questions, major risks, etc. that will then be broken down to develop your MAHD Task Backlog.

Developing a MAHD Task Backlog

IT'S TIME TO DEVELOP THE PRELIMINARY LIST OF BACKLOG TASKS

After reading about the previous four elements of the MAHD On-ramp, you may be thinking that these activities will take a long time. However, completing on-ramp activities should only take from a few days to a couple of weeks depending on the complexity of your project. This is still a fraction of the time that the typical product requirements phase of a waterfall project takes. While there are many details left undefined, you'll have a very clear picture of the vision, the customer, the high level roadmap and the overall project plan. You're now ready for the final on-ramp activity - developing the MAHD Task Backlog.



Task Backlog

As mentioned before, the backlog in Agile for SW methods is a combination of prioritized user stories, engineering tasks and features. For the MAHD Framework, the backlog is similar, but has distinct differences. A user story will not typically be a specific backlog item. A feature may be listed as a task, but generally the MAHD Task Backlog is a prioritized list of engineering and design tasks that are all directly related to user stories and product attributes.

To refer back to our fork lift example, the backlog might include:

1. Investigate open-source optical recognition solutions appropriate for fork lift applications.
2. Design new gearing to increase fork lift speed and accuracy.
3. Design new fork lift attachments to accommodate the largest number of material configuations.
4. Etc.

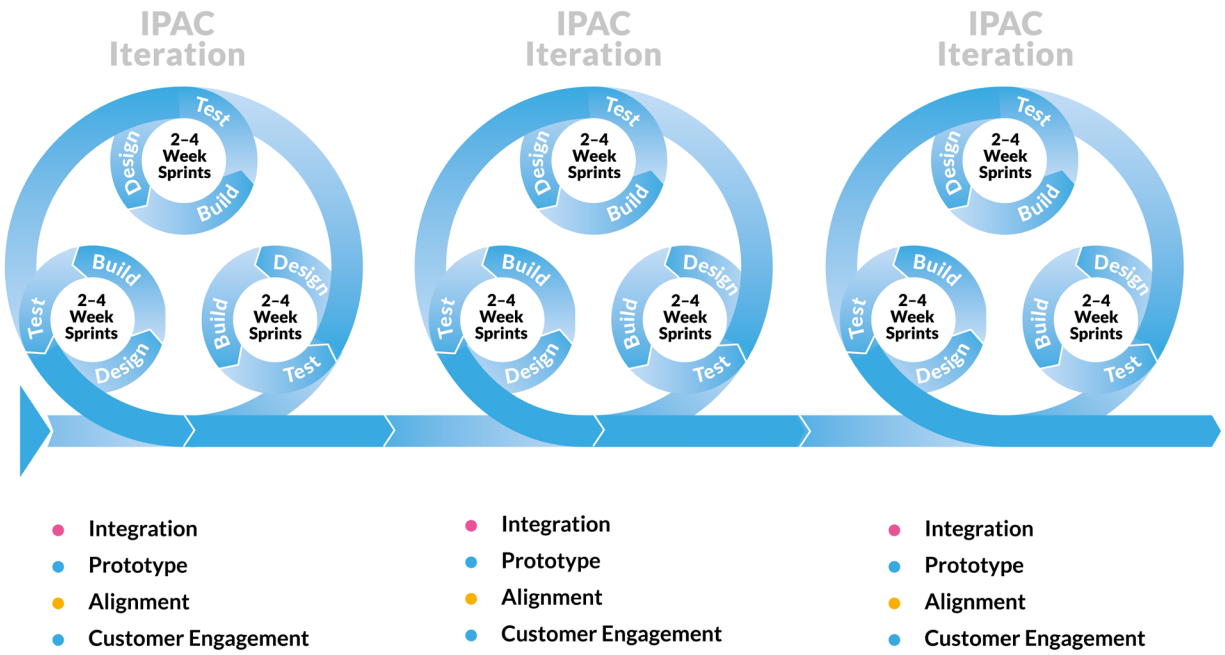
As the backlog is groomed (meaning clarified, estimated and the addition of acceptance criteria), you'll likely determine that many high-level tasks, such as the third one, "Design new fork lift attachments..." will need to be broken down further into tasks that can be accomplished in a single sprint. You could also make this task an "epic" and add tasks and sub-tasks such as:

- Identify the most common pallet configurations, dimensions and variations
- Identify Amazon's inventory sizes and configuations (assuming Amazon is a target customer)
- Etc.

Once your MAHD Task Backlog is ready (and it's never complete as you learn and add items), you can then begin planning your first sprint and you're off and running with Agile.

The Daily Grind: Planning and Executing Sprints

When most people think of "Agile," they imagine short development cycles where small teams commit to and deliver a set of features selected from a backlog. This provides the foundation of Agile methodology and executing MAHD sprints is similar. The only difference is that the development teams are selecting from backlog tasks that often look quite different than software features. Tasks may include specific features, but they are more likely to be technical investigations, design work, prototype building, integration of components, selection of a vendor, documentation for BOMs, etc. These tasks are aligned with the activities necessary for physical, mechanical and/or electronic designs and combine to build features and satisfy user stories.

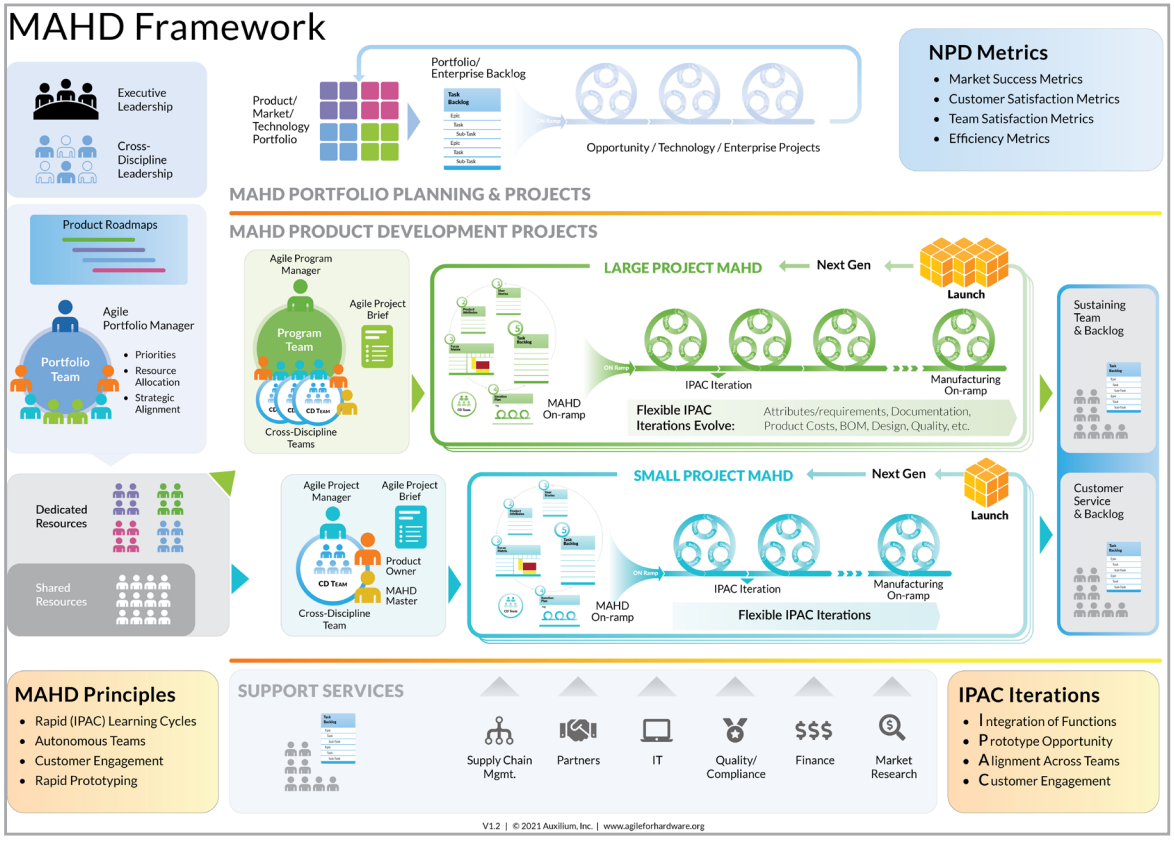


When planning for sprints, MAHD cross-functional teams will review the task backlog, clarify tasks and determine the number of tasks they can commit to during a two-to-four week sprint. As with any Agile process, the governance process is based on regular meetings where teams share their progress, groom tasks for the next sprint and validate their work. Teams may also choose to hold daily "stand-up meetings" typical of Agile SW practices or opt for semi-weekly or weekly meetings since progress for activities related to hardware are often not as granular as software tasks.

During sprint planning the team will also have an eye to the whole product (IPAC) iteration. Iterations are important to bring the components together and deliver prototypes that can be used for validation with customers. As sprints are completed, progress is tracked and the team will continue to improve their estimation skills and start showing real output months ahead of waterfall processes.

Scaling MAHD for Product Lines and Complex Products

The Complete MAHD Framework below shows the key elements for managing projects ranging from simple cost reductions to whole product portfolios. While the key roles and elements are typical of many new product organizations, the Framework is designed to allow you to adapt to your situation, culture and objectives.



While all projects will use the primary MAHD principles and methods, there are some additional elements to consider as you scale. A summary of these include:

1. **A MAHD team-of-teams approach –** With major projects, having one large MAHD team isn't often practical and you need mechanisms to manage multiple teams.
2. **New roles –** While each MAHD team will always have some form of Product Owner, Agile Project Manager and MAHD Master, additional roles are needed to manage larger projects, whole product lines and the overall portfolio.
3. **Multi-level Iteration Plans –** For large systems, you might even have multiple levels of Iteration Plans to manage each major workstream.

LEARN ABOUT THE COMPLETE MAHD FRAMEWORK BY VISITING: WWW.AGILEFORHWARE.ORG/INTERACTIVE-MAHD

The MAHD Difference

Can't We Just Use Scrum?

Scrum is the most commonly used Agile methodology for SW-based projects so we'll use this as the basis of comparison for each element of Agile vs. MAHD. The following table summarizes the Agile element and what's different for hardware.

Agile Element	Scrum for SW	MAHD
User Stories	Can be translated directly into tasks and backlog items.	Provide customer requirements at a system level, but typically cannot be translated directly into tasks.
Backlog	List of user stories, technical stories and epics. Constantly updated and prioritized.	List of project tasks derived from Iteration and sprint planning. Updated & prioritized each sprint.
Iterations	1-4 week sprints, each with code releases that can be demonstrated and tested.	2-4 week sprints grouped into whole product (IPAC) iterations to hit important learning milestones.
Requirements	Defined by user stories.	Defined by attributes, features, design targets and user stories.
Prototypes	Working software	Demonstrable prototype of varying sophistication
Releases	Working software that can be tested with direct user interaction.	Strategic prototypes that show demonstrable output to enable technical and market validation.
Deciders	Typically a SW-oriented product owner.	Typically a business-oriented product manager.
Process Owner	Scrum Master	Agile Project Manager
Team Orientation	SW Teams	Cross-functional Teams
Focus Matrix	Not used	A crucial planning activity to determine priorities, dependencies and prototype strategies.

Getting Started with MAHD

We've just touched the surface to fully describe the details of the MAHD Framework. If you'd like to get started applying Agile principles to your product development efforts, consider the following eight tips:

1. **Determine the fit** - Start with a clear understanding of your project goals to determine if Agile is right for you.

2. **Ensure you have top-down support** - If senior management does not understand or support Agile principles, it will cause a great deal of frustration that often leads your team back to waterfall techniques.

3. **Clarify roles and deciders** - You likely have titles like product managers, project managers and other roles. Determine if job definitions and responsibilities need to change to support Agile principles.

4. **Start slow and expect early hiccups** - If you've ever instituted a new process such as six sigma or phase-gate, you know it can take time and energy. Implementing Agile does take a couple of projects to become proficient. Stick with it.

5. **Don't skimp on the on-ramp** - Hardware requires up front planning to develop a clear vision and iteration plan before diving into sprint planning.

6. **Don't add tools too quickly** - Agile project management tools, such as Jira, can be a big aid, but learn Agile methods first and then add tools to improve efficiencies.

7. **Identify and train champions** - It takes leaders who have embraced Agile and are skilled in its usage. Identify a small group of evangelists who can lead the team.

8. **Don't skip customer interactions** - Agile requires direct feedback from customers. Many teams attempt to treat Agile as a pure development process, but to leverage its power, Agile must become a product success process.

With the right support, resources and coaching, these activities and tools will naturally lead to better NPD environments and long term market success.

ABOUT US

How Can We Help?

The MAHD framework was developed by Gary Hinkle and Dorian Simpson with input from dozens of companies ranging from furniture makers to medical equipment developers to address the needs of hardware development.

Having both been involved with product development for years, we have seen the challenges of waterfall-based NPD processes and how Agile can help. However, working with teams trying to implement Agile processes designed for SW development, we were determined to find a better way without throwing out the foundation that Agile methods provide.

The MAHD framework is available for all to use, build on and improve. We look forward to hearing from you and your experiences with Agile, waterfall and other processes.

To learn more, get involved, or just join our community for discussion, visit:

www.AgileforHardware.org

About Gary Hinkle

Electronics, mechanical and software engineering are all part of Gary's background, working in design, management and executive leadership of communication, industrial, telemetry, audio, avionics, computers, test & measurement, among other industries. Today, he's principal consultant at Auxilium, a company he founded to help engineering-oriented businesses increase productivity.

Contact Gary

P: 971-222-6234
E: gary@auxilium-inc.com

About Dorian Simpson

Dorian Simpson is an innovation, product management and agile consultant, trainer and speaker as well as author of *The Savvy Corporate Innovator*. Companies he's worked with include ABB, Tyco, Owens Corning, FEI, Freightliner and dozens of others. Before consulting, Dorian held senior positions at Motorola and AT&T in product management, sales, marketing, business development and engineering.

Contact Dorian

P: 971-235-4905
E: dorian@auxilium-inc.com

Auxilium

Learn more about our agile training and consulting services at: www.auxilium-inc.com