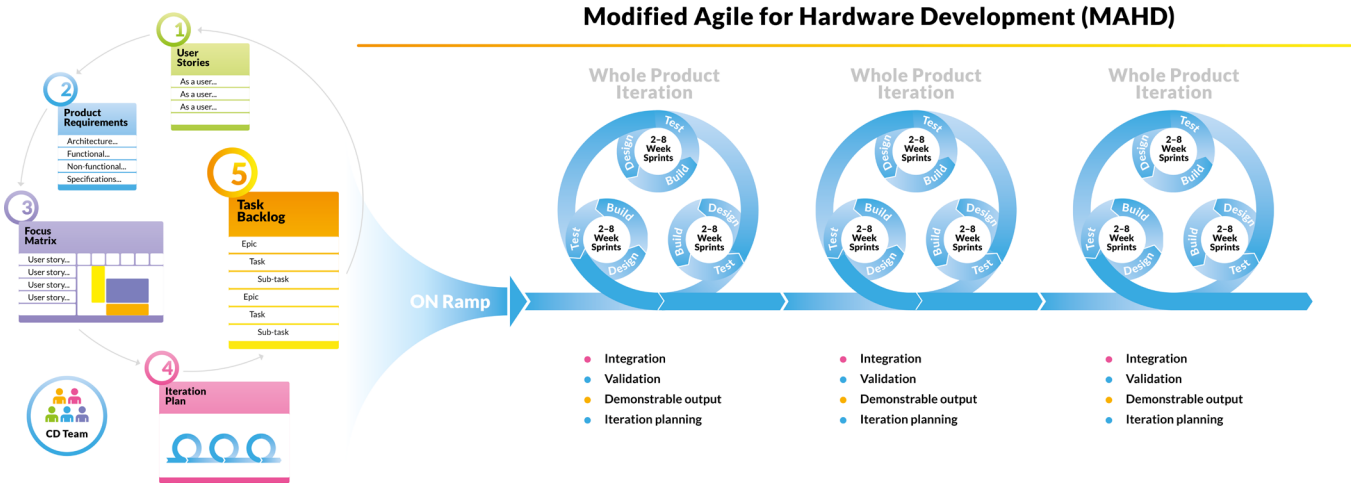
WWW.AGILEFORHARDWARE.ORG



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 this Ebook?

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.*

Letters from the Authors

Dorian Simpson
Kingsley Institute
MAHD Evangelist

I've spent my career in product development starting as an engineer for IBM and on to management while developing both hardware and software products for a wide range of companies. Early experiences with phase-gate systems were frustrating and while good-intentioned, never seemed to solve real business needs. I consistently strived to bring customers into the development process and work with R&D teams to stay flexible as we learned, but we didn't always have the tools and language to effectively do this since for most of my career Agile wasn't even an understood term. As development methods matured and Lean, Agile and Design Thinking took hold, I was excited to see more people embrace these principles. Today there is still a gap as we bring the methods together for hardware development. My hope is that the MAHD Framework will help fill this gap as we continue to make better processes more accessible and easier to apply in all situations.

Gary Hinkle
Auxilium Inc.
MAHD Evangelist

I've spent my career in product development and have always had a passion to make new product development processes easier, more efficient, less siloed and more powerful. Unfortunately, early experiences with phase-gate systems were frustrating and while good-intentioned, never seemed to solve real business needs. I consistently strived to bring customers into the development loop and work with R&D groups to stay flexible as we learned, but we didn't always have the tools and language to communicate how to do this. At the time, Agile wasn't a term. As development methods mature and Lean, Agile and Design thinking take hold, I was excited to see more and more people adopt the principles, but there is a still a gap as we bring the methods together for hardware development. I hope we can help fill this gap with the MAHD Framework and continue to evolve Agile, lean and design thinking to be easier and more accessible.

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 were just too great to accurately write requirements. 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 goodness? Yes. The philosophy is sound, but the application of Agile requires significant changes to support the needs of hardware products where making changes are costly, partial products are difficult to test with real customers and schedules are demanded by management. This led the need to develop the Modified Agile for Hardware Development (MAHD) Framework — an open-source initiative to embrace the principles 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

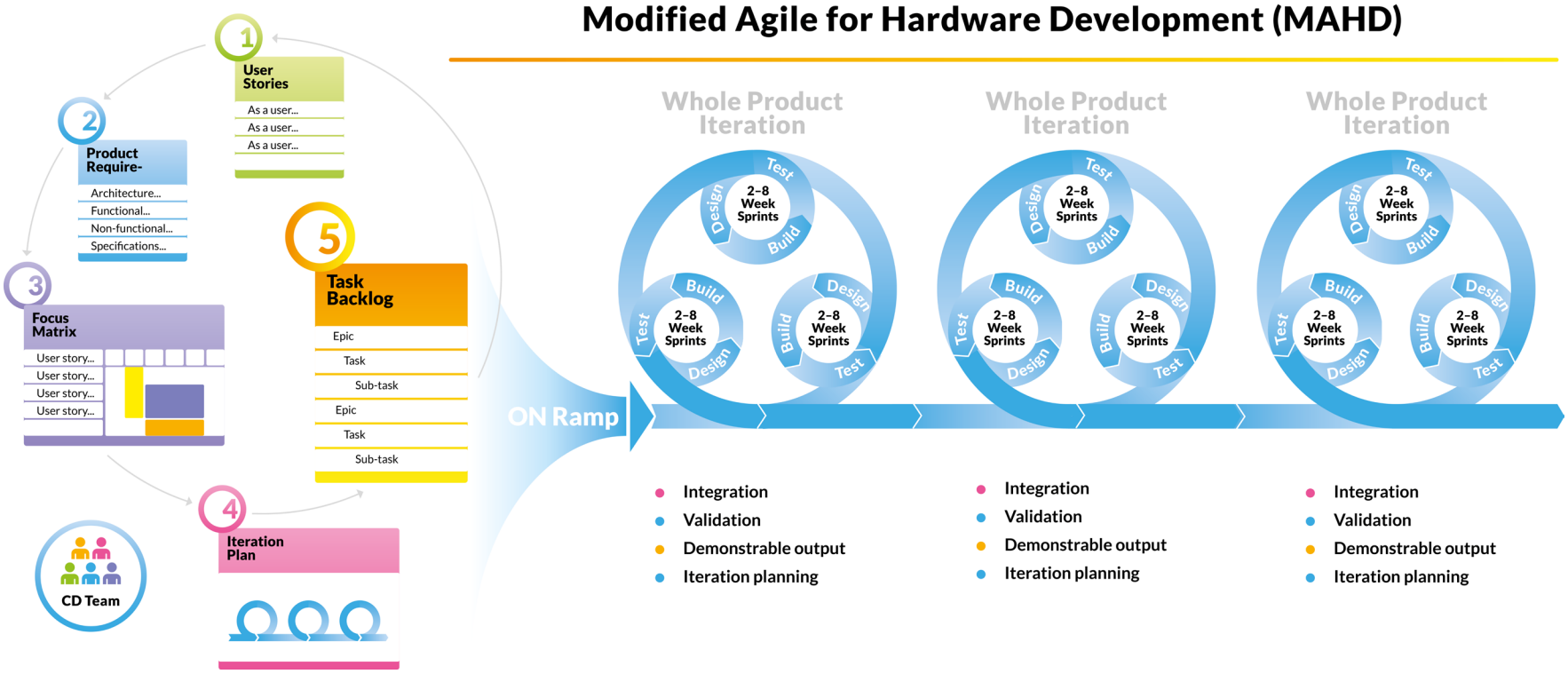
Just as with Agile for SW methods, the MAHD Framework has at its core the concept of short development and learning cycles. But as the model shows, the MAHD Framework includes two levels of cycles that must be considered to accommodate the needs of hardware development.

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.

Each of these are described in more detail below. As the MAHD Framework evolves and our community grows, we will continue to refine the framework, add case studies and provide detailed examples. To see the latest updates, visit: **www.agileforhardware.org**

For those familiar with Agile for SW methods, Appendix A has a more detailed comparison of the MAHD Framework versus Scrum, the most commonly used Agile for SW methodology.



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.

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 needs to be considered before diving into the work itself.

As we'll describe, the MAHD On-ramp may take 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 reconsidering their use when defining physical products.
2. **Product Requirements** - While Agile for software does away with requirements (which are replaced with user stories), requirements 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 Plan** - Similar to a SW sprint, but quite different. At two levels, MAHD Iteration Planning creates 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 a lot like a waterfall-oriented process, this doesn't mean the team writes a detailed product requirements document, a complex Gantt chart or get 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."***

As a software developer, you 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 cargo so that I can save time moving inventory."***

As a developer of hardware, do you know what to do? Probably not. There are too many facets of the problem that need solved. 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 NECESSARY, 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 requirements and the focus matrix.

Requirements: Don't Throw Away Those PRDs

HOW TO CONSIDER PRODUCT REQUIREMENTS IN AN AGILE WORLD

Just the term "requirements" is almost anathema to Agile purists. We 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 as well as 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 cargo 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 Requirements

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 requirements is important as we'll see in the next step. Keep in mind that "requirements" as defined in the on-ramp aren't rigid specifications, but really just the ideas that act as a starting point to seed the Agile process and guide it in the right direction. Each requirement will get refined into specific features, functions and attributes that satisfy customer needs (the user stories) as the Agile process moves forward.

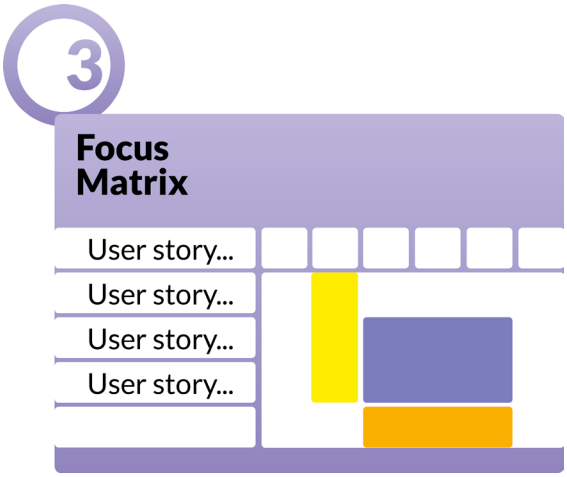
To summarize, without describing the product in more detail during upfront planning it will be difficult to create the backlog of tasks as well as to identify areas of functionality that will be used for iteration planning, prototyping and the focus of innovation efforts.

Driving Agile Priorities with a Focus Matrix

CONNECTING USER STORIES, REQUIREMENTS AND ITERATIONS

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 features and product attributes and various product features 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 to 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 features and functional requirements. The relationships between these factors become the basis for the high-level 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 features 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 and how to get customer feedback as well as identifying dependencies.

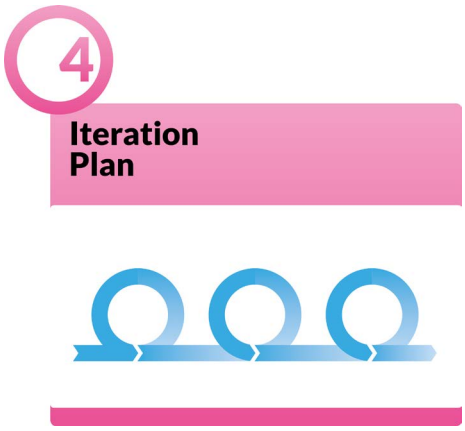
Successfully working through the Focus Matrix leads to the next MAHD On-ramp activity, iteration planning. The result of which becomes the high-level project plan that is necessary for managing risk, schedules, resources and tasks.

Iteration Planning in a MAHD World

CONNECTING USER STORIES, REQUIREMENTS AND ITERATIONS

As mentioned early, at the core of any Agile methodology are iterative development and learning cycles. For Scrum-based Agile for SW, an iteration is called a sprint and lasts from 1-to-4 weeks and the result of a sprint is working software that can be demonstrated to users. The MAHD Framework redefines this concept and considers two levels of iterations. At the high level are "whole product iterations." These iterations lead to larger deliverables where components of the solution are integrated and a key result is a demonstrable prototype that can be viewed and validated with real end customers.

As the MAHD Framework shows on page six, there are also a series of shorter execution cycles within iterations, these are 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, answer three questions:

1. Which areas have you identified that are high risk, but important to get right for customers?
2. In what order should the attributes (and associated user stories) be developed to account for dependencies?
3. How will you envision the prototype plan coming together at different levels of sophistication in order to gain real customer insight?

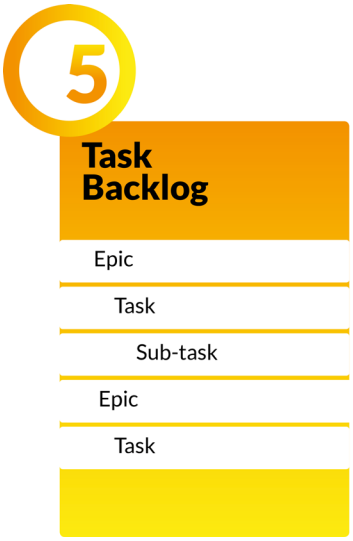
Back to our fork lift example. Let's assume you decide that determining the orientation of inventory is crucial to satisfying a high priority user story. You then plan to focus on developing a solution and early prototype (that may or may not include optical recognition) in an early iteration in order 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 sprints, but only identifying the large work buckets, dependencies and planned prototypes 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 the on-ramp activities should only take one to three weeks depending on the complexity of your product. 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.

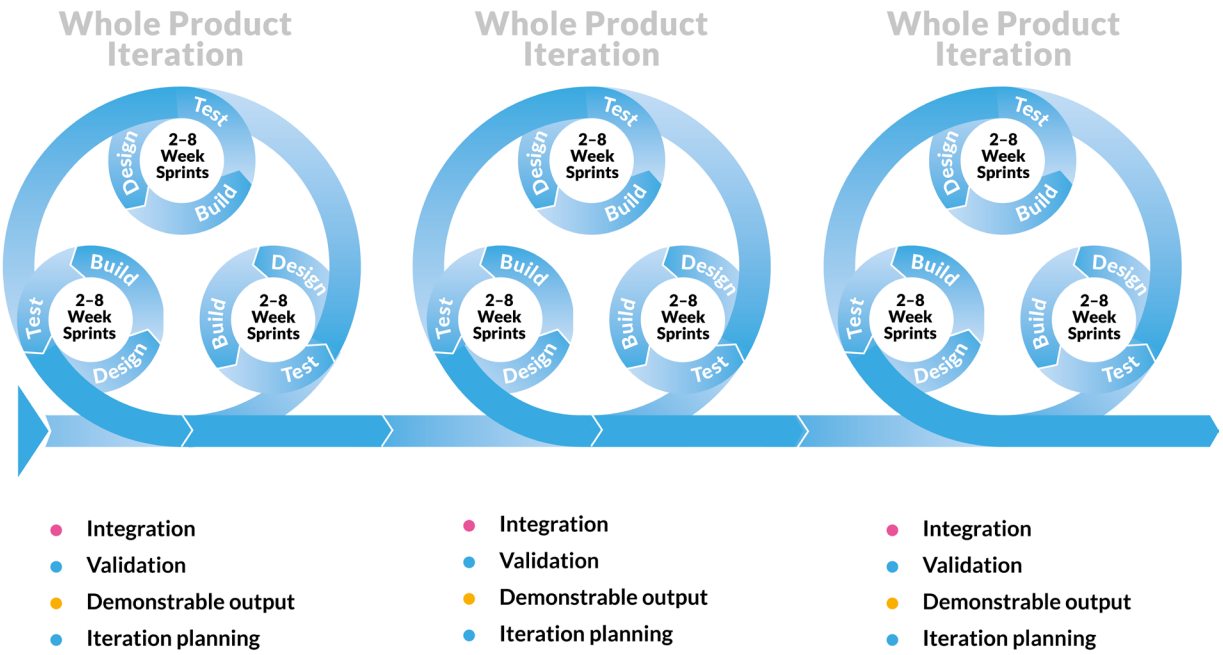
As the backlog is "groomed" (meaning clarified, estimated and the addition of acceptance criteria), you'll likely determine that many high-level tasks, such as the third one, "Design new fork lift attachments..." will need to be broken down further into tasks that can be accomplished in a single sprint.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

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

Once your MAHD Task Backlog is complete (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-eight 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 will also choose to hold daily "stand-up meetings" typical of Agile SW practices or opt for bi-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 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 continute to improve their estimation skills and start showing real output months ahead of waterfall processes.

Getting Started with MAHD

We know 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.

The MAHD Difference

Can't We Just Use Scrum?

Scrum is the most commonly used for 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, but typically cannot be translated directly into tasks.
Backlog	List of user stories, technical stories and epics. Constantly updated and prioritized.	List of tasks derived from MAHD On-ramp planning. Constantly updated and prioritized.
Iterations	1-4 week sprints, each with code releases that can be demonstrated and tested.	2-8 week sprints grouped into whole product iterations to test key features and specifications.
Requirements	Defined by user stories.	Defined by features, design targets and user stories.
Prototypes	Working software	Demonstrable physical prototype
Releases	Working software that can be tested with direct user interaction.	Working prototypes that show demonstrable output for technical and user validation.
Deciders	Typically a SW-oriented product owner.	Typically a business-oriented product manager.
Process Owner	Scrum Master	Project Manager
Team Orientation	SW Teams	Cross-functional Teams
Focus Matrix	Not used	A crucial planning exercise to determine iterations, dependencies and prototypes.

ABOUT US

How Can We Help?

The MAHD framework was developed by Gary Hinkle and Dorian Simpson 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 an open-source process, 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 Hinkle'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

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

About Dorian Simpson

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

Contact Dorian

W: www.KingsleyInst.com
P: 971-235-4905
E: dorian@kingsleyinst.com

Kingsley Institute
for Strategy and Innovation

Auxilium