WWW.AGILEFORHARDWARE.ORG

MAHD

Step-by-Step Toolkit

Applying Agile Principles to Physical Products
When Making Changes Is Difficult

By Dorian Simpson and Gary Hinkle

CONTENTS

What's Inside this Ebook?

Letter From the Authors 3

An Introduction to MAHD 4

The MAHD On-ramp 10

- User Stories 11
- Product Attributes 14
- The Focus Matrix 17
- Iteration Planning 20
- Task Backlog 24

Planning Sprints 27

Gettng Started with MAHD 32

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 provide step-by--step guides for each element*
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

Since I started leading engineering projects in the 1990's, I had a natural desire to learn about best practices and improve processes. In my quest for wanting to find a better way, I discovered that Agile methods are the best fit for most projects – even before Agile became popular in software development. Unfortunately, not every decision-maker around me has understood Agile and the benefits of speed, efficiency and focus that are so valuable to hardware-oriented businesses. While I have diligently worked to influence other leaders to adopt Agile thinking and methods, the lack of knowledge and experience in industry are significant barriers. Building awareness of the MAHD Framework and ultimately establishing MAHD as a practical standard will hopefully make it easier for organizations to implement Agile methods and realize their tremendous business value.

Is Agile Right for Hardware?

INTRODUCTION

Agile methods have taken over the software (SW) industry. Developers and leaders have discovered that traditional waterfall processes don't work because the upfront unknowns are usually too significant 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 (HW) 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. Note that when we use the term "hardware," this could be any project that goes beyond software and includes electronic, mechanical and other elements requiring a cross-discipline team.

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 terms 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 diagram on the following page, the MAHD Framework 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:

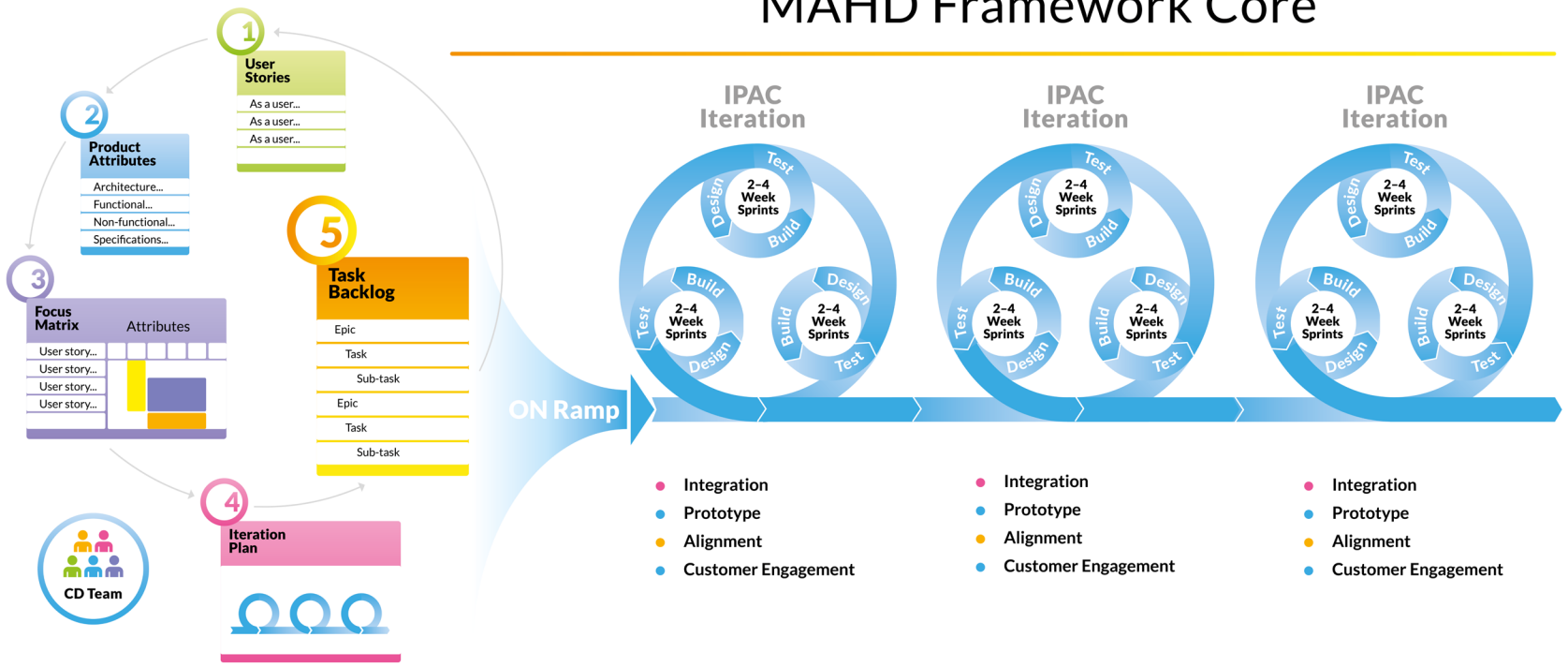
The On-ramp
One of the basic principles of Agile is to develop very little formal documentation, allowing the team to focus on the most valuable activities. This is true of Agile for hardware methods also, but the nature of physical products requires additional upfront planning steps and often more documentation.

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 iteration cycles 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**

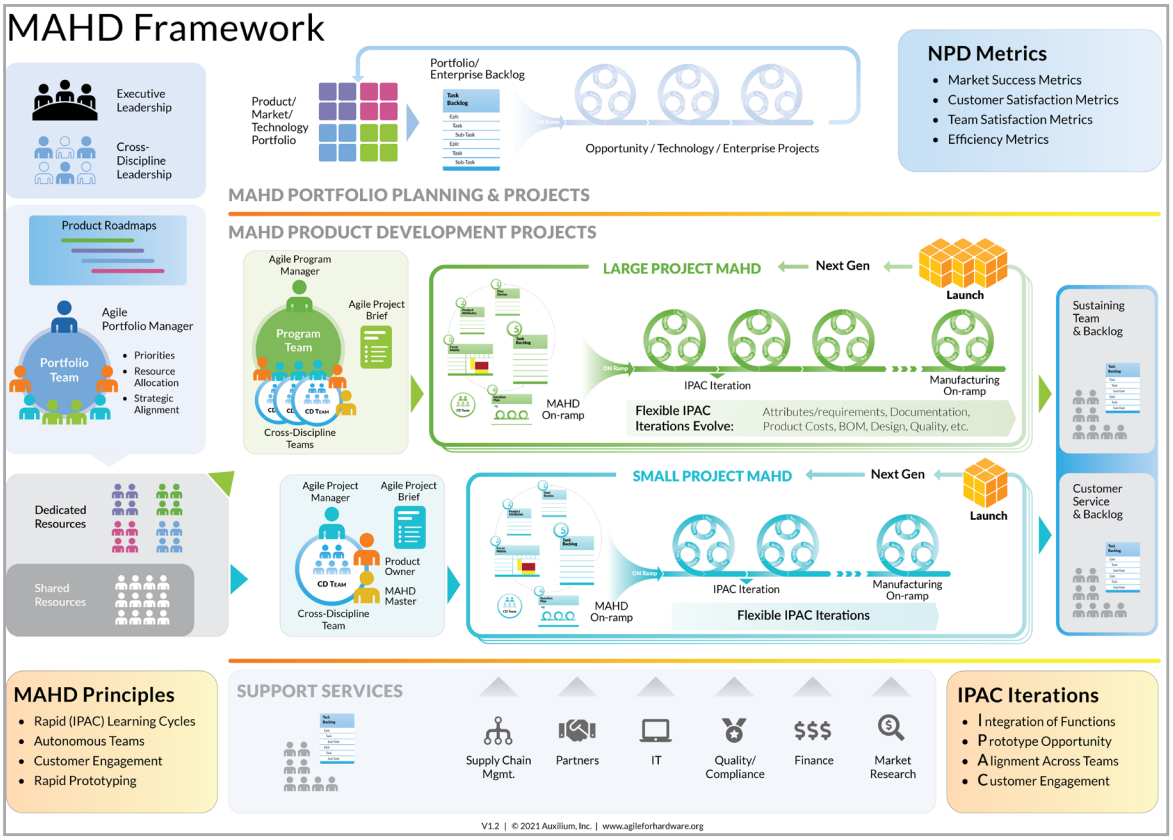
For those familiar with Agile for SW methods, Appendix A has a simple comparison of the MAHD Framework versus Scrum, the most commonly used Agile for SW methodology.



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.

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

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 Project 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):

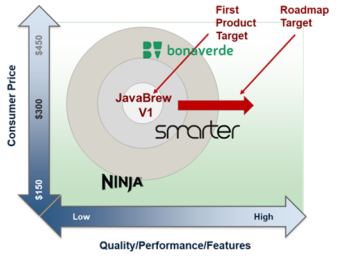
- Attractive design: Pleasing, fit with decor, clean, modern
- Quality of coffee: Taste, consistency, flexibility
- Long term experience: Maintainable, functional, durable
- Smart: Easy, cool, intuitive, new use cases

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

MAHD ROLES: Modified Roles Are Essential for Success

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 below shows, the roles are similar, but there are a couple of important distinctions. The first is the "decider" role. For software, the decider is 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 software development. For HW this role will often be filled by a Product Manager. This person must understand and consider the whole product and make decisions that may have cross-discipline development, product and market implications.

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

Deciders

Those who define the product and have responsibility for tough priority decisions.

SW: The Product Owner

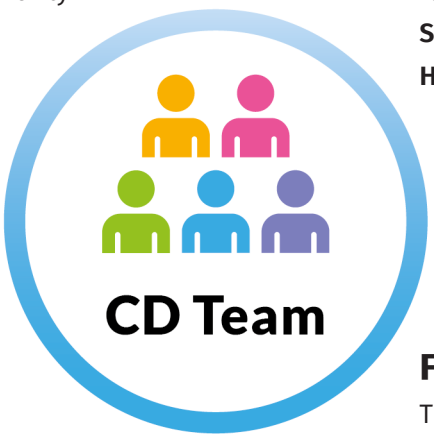
HW: The Product Manager

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

Preparing for Agile Success with the On-ramp

HARDWARE REQUIRES DEEPER UPFRONT PLANNING THAN SOFTWARE

One of the major challenges that the MAHD Framework addresses for hardware is the need to think about the whole project before starting with Agile execution. Designs, architecture, dependencies, high-level iterations and even the schedule need to be considered before diving in. The steps to do this planning are included in the MAHD On-ramp.

As we'll describe, executing the on-ramp steps may take a few days or weeks, but the results are well worth the effort in setting up the team for success. Many elements are similar to those found in Agile for SW, but each element is modified. We'll also introduce a new element — the MAHD Focus Matrix — specifically designed to support hardware development efforts. The following pages will describe each of these five elements in more depth:

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 Attributes** - While Agile for SW does away with requirements (which are replaced with user stories), it's still necessary to describe the major attributes of the product before beginning Agile execution.
3. **Focus Matrix** - This is a new element to Agile introduced in the MAHD Framework. It builds a necessary bridge between user stories and product attributes to drive iteration planning, Agile thinking, innovation and development focus.

4. **Iteration Plan** - Similar to a SW version plan and sprints, but quite different. MAHD uses two different levels of iterations. MAHD whole product iterations create the game plan for success while MAHD sprints drive the execution details.
5. **Task Backlog** - Similar to a SW backlog, but again... different. The MAHD Task Backlog still includes prioritized tasks to be executed, but you won't find user stories here.

While some of these activities may look like a waterfall-oriented process, the MAHD On-ramp does not require the team to write detailed product requirements or develop a complex Gantt chart. It does require 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 the stories provide the backlog items, are groomed directly into features and sprint tasks, and form the basis of 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."***

For software, this story is straightforward. Developers would know what to design and likely how to implement it as a feature to satisfy the story.

Now let's try it for hardware. Let's assume you're planning to develop a coffee-making appliance and you write this user story:

> ***"As a user I want to be able to modify the coffee strength to accommodate different tastes in my house."***

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 rate of drip, the amount of coffee used, the interface that gives customers control, etc. 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 user stories are still 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 MAHD User Stories are written, it takes several more steps to identify the specific backlog tasks. User stories are then referred to often in order to ensure each task, feature and product attribute is focused on satifisfying the most important user stories and making tough tradeoff decisions.

In an upcoming step, you'll build a relationship between these stories and anticipated product attributes (features, functions and technology) to develop a task backlog. Before you can do this, we'll need to consider how to consider "product requirements" in the MAHD Framework.

USER STORIES: Step-by-Step

DESCRIPTION:

User stories generally follow the format:

As a _____, I want to _____ so that I can _____. The tool to document user stories can be as simple as an excel spreadsheet or even sticky notes on a whiteboard. They should be categorized into themes and clearly prioritized based on real customer insight.

STEP-BY-STEP:

1. Clarify who your customer is using customer personas.
2. Identify high-level customer themes (can be before or after writing user stories).
3. Add user stories related to each theme that fulfill your vision for the project from customers' perspectives.
4. Add preliminary priorities to each user story.
5. Add notes to clarify and links to designs or related documents.
6. Share your user stories with all stakeholders and explain the overall customer experience.
7. Update the stories as you learn more from real customers.

WHAT CAN GO WRONG?

1. Writing user stories only to define internal tasks.
2. Using user stories to describe features (E.g. As a user, I want an "On button."
3. Prioritization is unclear and/or not based on customer needs.
4. Not updating the backlog on a regular basis to match changing user requirements.
5. Using personal opinions to set priorities rather than real customer insight.

TIPS FOR SUCCESS:

1. Don't develop user stories until you've developed a clear understanding of the customer.
2. Work with your cross-functional team to clarify, refine and prioritize the stories.
3. Question every user story to ensure it's describing a real customer need and not just a feature.
4. Add preliminary acceptance criteria to top user stories before grooming.
5. Always ask yourself, "Would real customers agree this is a good user story?"

WHO OWNS THIS ACTIVITY?

1. The **Product Manager** - Writes user stories and guides the team to understand their relationship to features and technical attributes.

WHEN DO YOU DO IT?

Required to initiate an Agile project and updated before each sprint and iteration planning session.

USER STORIES: Template and Example

OVERVIEW

User stories should provide a complete view of the product from a customer perspective. Below shows a sample of user stories that might define the coffee maker. Notice that there are no specific features or technology included. Customers don't care about features, they only care what the product will do for them, and this is what user stories attempt to define.

Once user stories are understood, acceptance criteria should also be added, as well as how you'll get validation for the user story. E.g. "I want an attractive appliance... " might not be satisfied until several prototypes have been developed and customer focus groups readily agree you've developed an attractive appliance.

Category	User Story (As a user...)	Pty.
Design	I want an attractive appliance so that it looks good on my counter	High
	I want a choice in colors so that it matches my tastes and decor	High
	I want to easily clean the applicance, or better, not have to clean it at all	Med.
	I want to setup the appliance without reading a manual to save time and stress	High
	I don't want to spill water when I add water to make coffee so I don't make a mess	Med.
	I want it to use standard filters... Or no filter, so that I can save time and money	Med.
Making Coffee	I want to easily control the appliance from anywhere to save me time and stress	High
	I want to make anywhere from one cup to many cups based on how much I need at the moment and not waste coffee	High
	I want to control the strength of the coffee for different tastes	High
	I want to conrol the timing so I can have coffee exactly when I want it	Med.
	I want the coffee to stay hot after it's ready so that I can enjoy it	Med.
	I want to control the maker in any lighting so I can get coffee in the dark	Low

Product Attributes: Not Quite Requirements

HOW TO CONSIDER 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 and these can be described as user stories. Hardware is similar... but different. Physical products often have subsystems, physical attributes inherent to the product, required components, target specifications, standard interfaces, etc. Certainly we could describe all of these details through user stories, but this approach is tedious to document, leads the team back to describing features and specifications (only in a different format), and reduces the real purpose of user stories, which is to describe the real needs of customers.

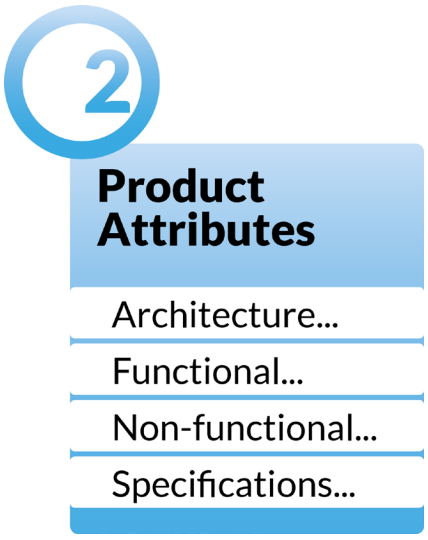
For example, if we consider our coffee maker user story again:

> ***"As a user I want to be able to modify the coffee strength to accommodate different tastes in my house."***

This story describes the real customer need and we could write more user stories to enumerate the specifics of the features and functions to satisfy this story, such as:

> ***"As a user, I want to control the taste from very weak to very strong with at least five different levels of strength."***

However, this is not really a user story. It is closer to being a target specification or perhaps part of the acceptance criteria of the user story.



Product Attributes

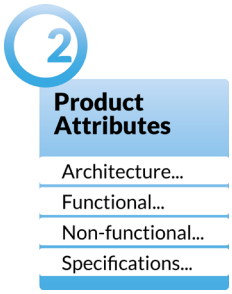
In the MAHD Framework, understanding and documenting the major product attributes is an important step as you'll soon see. We are not advocating that MAHD practitioners write detailed product requirement documents that are associated with waterfall processes, but instead, to document the range of anticipated attributes, describe the rough architecture and list any real requirements such as industry standards, certification needs, cost targets, standard components, etc. We'll need these to develop an iteration plan and task backlog as well as for using the Focus Matrix to identify important areas of risk, seek places to innovate and create a prototype plan.

Documenting the major product attributes and hard requirements will act as a starting point to seed the Agile process and lead to the task backlog. As the MAHD process moves forward, each product attribute will get refined into specific features, functions and specfications that satisfy customer needs (user stories.)

PRODUCT ATTRIBUTES: Step-by-Step

DESCRIPTION:

As part of the MAHD On-ramp, the team will identify the major attributes of the product to begin the process of developing the backlog of tasks. Product attributes are not detailed requirements you'd find in a waterfall Product Requirements Document (PRD), but does include buckets of functionality, major components or constraints that describe the overall product.



STEP-BY-STEP:

1. Use sticky notes to start or document the attributes and requirements using a spreadsheet.
2. Identify the major functional areas of the product including subsystems.
3. Categorize attributes and requirements into logical buckets.
4. Identify which functionality or components are fixed and unlikely to change.
5. Consider constraints as requirements, such as target cost, standards and certification needs.
6. Review as a cross-discipline team to clarify and add details.
7. Identify which attributes have associated major risk and/or dependencies.

WHAT CAN GO WRONG?

1. The requirements are too detailed and look too much like a "PRD" or specifications.
2. No indication of what is fixed and where there is room to innovate.
3. There is no understanding of business requirements such as cost, price or launch targets.
4. The requirements have not been reviewed with all stakeholders (ideally they are developed as a cross-discipline team.)

TIPS FOR SUCCESS:

1. Work with your cross-discipline team to clarify the attributes and identify gaps.
2. Include designers in the agile process as early as possible.
3. Always ask, "Is this set in stone? Or do we have flexibiity to innovate here?"
4. Attach pictures or links to anything that helps communicate the attributes.
5. Don't lock yourself into detailed decisions too fast - allow "room to innovate."

WHO OWNS THIS ACTIVITY?

1. The **TEAM** - A Product Manager or Chief Engineer often develops an initial draft. However, after the initial plan, the APM is typically responsible for updating.

WHEN DO YOU DO IT?

Required to initiate an agile project and updated before each sprint and iteration planning session as details are defined.

Product Attributes: Template and Example

OVERVIEW

Below shows a sample of product attributes that begin to define the coffee maker. While some may be specific and fixed based on real business or technical requirements, many other attributes will be "To Be Determined." These will need further investigation that will be completed as tasks in upcoming sprints.

For example, note the attribute of "Programming Buttons" below. A typical waterfall PRD would list the exact buttons along with their functionality. For Agile, we might assume the coffee maker has buttons, but we don't know how many, what kinds, or if we need buttons at all until the coffee maker is more fully defined. We will learn this as the Agile process progresses.

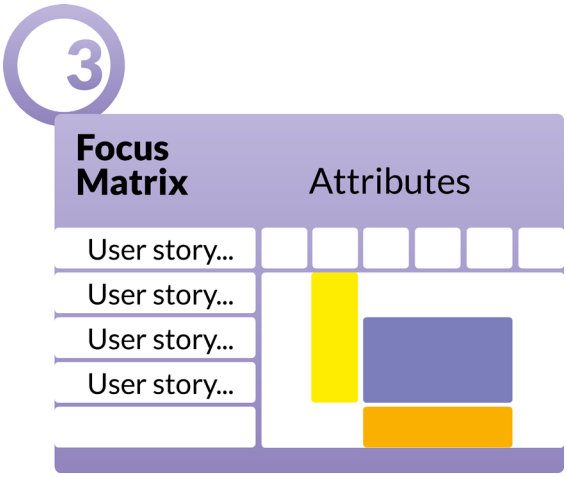
Category	Attribute	Discussion/Details
Physical Interface	Interface technology	Need "smart" decisions first
	Programming buttons	TBD - after ID decided
	On/Off mechanism	TBD - Voice?
	Temperature control	TBD - after ID decided
Smart Features	Voice Control	Alexa integration TBD
	Smart device interface	Research early!
Physical Design	Color Choices	Research with marketing - Popular?
	Physical material/texture	Options? Research - TBD
	Attractive Industrial Design	TBD
	Coffee Grinding Features	Use standard from P-020 series?
	Water access mechanism	Research - P-020 unpopular
	Flexble strength control	Research early! Many unknowns.
	Filter method	Use industry standard
	Thermal carafe design	Similar to P-020 - Should we modify?

Driving Agile Priorities with a Focus Matrix

BUILDING CONNECTIONS BETWEEN USER STORIES AND PRODUCT 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 MAHD planning for a 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 (attributes). 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 product attributes we developed in the last step. The relationships between these factors become the basis for the high-level iteration planning, prototype plans and dependency identification.

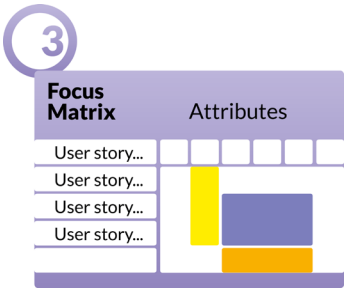
To complete the matrix, the team would identify the highest priority user stories and the attributes that will lead to their satisfaction. The results will provide the guide to a plan of execution for how to prototype the attributes and how to get customer feedback as well as which attributes have dependencies. Consider our coffee maker example. The user story, "…. ***be able to modify the coffee strength…***" would appear on the left and might be satisfied by a range of attributes that are organized on the top, such as: the coffee grinding mechanism, the filter method and of course the flexible strength control mechanism. This will likely need a prototype to fully define.

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.

FOCUS MATRIX: Step-by-Step

DESCRIPTION:

The Focus Matrix looks at important relationships between user stories and product attributes. You're looking for high areas of importance and where you'll need to apply more attention (innovation) and set priorities as well as begin to think about your prototype plan and how you'll get customer feedback.



STEP-BY-STEP:

1. Gather the team in front of a large whiteboard or wall.
2. List the user stories on the left, ensuring you have clearly prioritized and understood them.
3. List the relevant product attributes at the top.
4. Break into groups, or as a team, identify important relationships - "Considering the important user stories, which attributes should we focus on to give us an advantage or reduce our risk?"
5. Consider the prototype plan. "What are the levels of prototype we should develop to validate this combination? (Both technically and through customer validation)."
6. Maintain the results in a central location for iteration and sprint reviews.

WHAT CAN GO WRONG?

1. Not taking the time to conduct this activity.
2. Not including the whole cross-discipline team.
3. Limiting thinking to assume you can't change "requirements."
4. Having too many areas of focus (I.e. if everything is important, nothing is).
5. Not having a good facilitator to lead this activity.

TIPS FOR SUCCESS:

1. If this activity is difficult, consider reviewing the user stories and attributes. Have user stories really described what customers desire? Are the attributes too detailed?
2. Maintain an open mind. This activity should enable new ways to satisfy customer needs. Always ask yourself, "Will improving this attribute help customers in imporant areas?"
3. Let the prototype and integration plan drive iteration durations.

WHO OWNS THIS ACTIVITY?

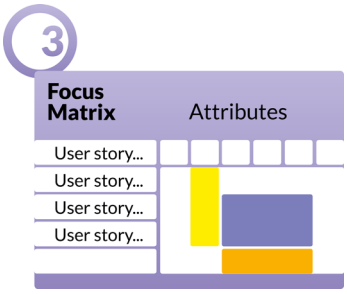
1. The **Team** - This activity must be completed as a cross-discipline team. The APM or other facilitator should lead. The APM will update as needed.

WHEN DO YOU DO IT?

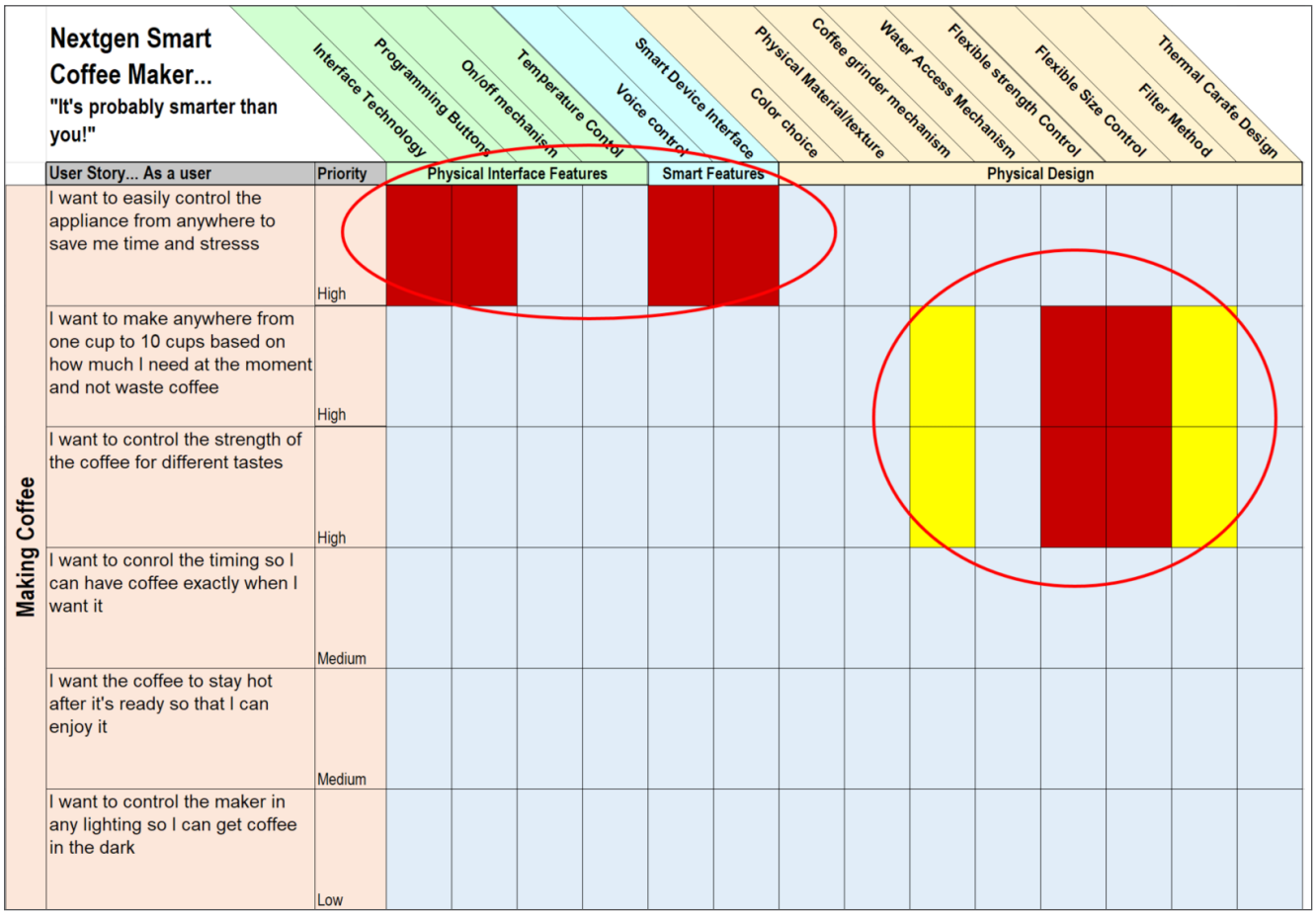
Required to initiate an agile project and reviewed/updated with each iteration planning session.

FOCUS MATRIX: Template and Example

DISCUSSION:



Below shows a sample of the Focus Matrix for our coffee maker example. The team reviews the top user stories and identifies the attributes that most contribute to their success. They determine they will need to develop an early prototype in two areas; 1) To satisfy the top user story, "... to easily control the device from anywhere...," they will need to figure out the smart interface and its interaction with the physical interface. and 2) They will need to figure out how the strength and size mechanisms, along with the filter and grinding mechanisms, will work together to give the user flexible size and strength. These will both be important areas to develop as high priority tasks in the backlog as well as in the iteration plan.

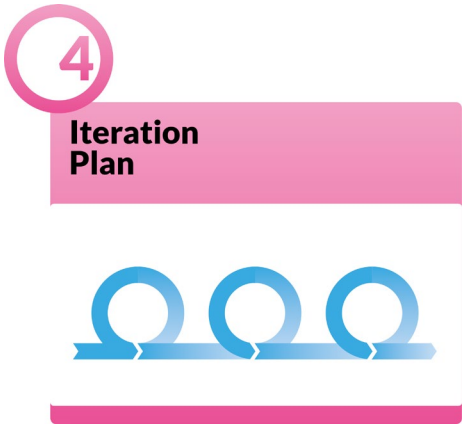


Iteration Planning in a MAHD World

THE ROADMAP FOR AGILE SUCCESS... AND A PLAN MANAGEMENT NEEDS TO SEE

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. The result of a SW 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 or "IPAC iterations." These iterations lead to larger deliverables where components of the solution are integrated and key results include demonstrable prototypes that can be viewed and validated with real customers. At the second level there are also a series of shorter execution sprint cycles within iterations. 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. However, working toward prototypes at the whole product iteration level 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 with the right milestones attached to each iteration, answer four questions:

1. **Integration**: What are the integration points where software, electrical, mechanical, etc. must come together for technical or market validation?
2. **Prototype**: How will you envision the prototype plan coming together at different levels of sophistication in order to gain learning?
3. **Alignment**: When and how will you get other functions and stakeholders (including executives) aligned with progress?
4. **Customer Engagement**: When and how will you engage with real customers to get feedback and key questions answered?

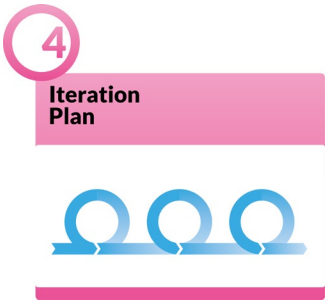
Back to our coffee maker example. Let's assume the team decides that defining the variable strength coffee features correctly is crucial to satisfying a high priority user story. During iteration planning the team would focus early on defining a solution and developing a prototype in order to validate the technology and determine if customers will accept the results. If the team decides this is not be possible until the third iteration, they might decide the second iteration will include an animated video to demonstrate the solution to 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, integration points and planned prototypes that will then be broken down to develop your MAHD Task Backlog.

ITERATION PLANNING: Step-by-Step

DESCRIPTION:

Iteration planning is where your plan comes together. You'll estimate the overall schedule, determine integration points, identify major dependencies, establish milestones and define prototype deliverables. While you'll end with a plan at the end of the MAHD On-ramp, it will be consistently updated through sprint and iteration planning.



STEP-BY-STEP:

1. Using the Focus Matrix as a starting point, work as a team to identify the major milestones. These should include prototypes, major subsystems and integration points.
2. Estimate the duration of iterations necessary to complete the project.
3. Determine the duration of a sprint by targeting 3 to 4 sprints per iteration (ideally sprints are 2 to 4 weeks in duration).
4. Add customer feedback goals for each iteration. What major questions do you need to answer?
5. Look for areas of high risk and develop a risk mitigation plan.
6. Document the plan and ensure everyone has visibility to it. Track during sprint planning.

WHAT CAN GO WRONG?

1. Do not attempt to develop a detailed Gantt chart — stay at a higher deliverable level.
2. Not including the whole cross-discipline team or building concensus.
3. Details are important, but every decision does not need to be made at this point.
4. This activity requires a strong facilitator... is yours up to the task?
5. Not updating the iteration plan as you learn.

TIPS FOR SUCCESS:

1. While it is good practice to use consistent iteration durations to build momentum and predictability, it's ok to have iterations of different lengths to accommodate the reality of hardware development such as waiting for board spins, tooling, etc.
2. Try not to lose focus on the iteration plan as the team begins to focus on sprint execution.
3. Don't forget the time it takes to engage with customers in preparation for the next iteration.

WHO OWNS THIS ACTIVITY?

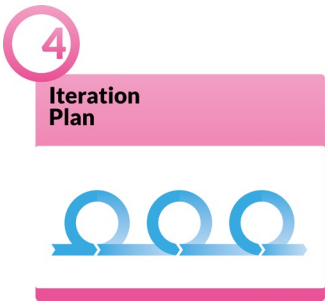
1. The **Team** - This activity must be completed as a cross-discipline team. The APM should facilitate and will update as needed.

WHEN DO YOU DO IT?

Required to initiate an agile project and updated before each sprint and iteration planning session.

ITERATION PLANNING: Template and Example

OVERVIEW



Below shows a sample of an iteration plan for our coffee maker example. Major iteration milestones are developed using the levels of prototypes and integration points as a guide. The team has a target production date that also forces them to work toward a fixed schedule. The means that the schedule has priority over the tradeoff of resources and scope. As sprints and iterations are completed, if progress is slower than expected, the team will need to make tough decisions to simplify the product or find resources to complete the work on time.

For example, in iteration 3, the team will test their concept for the flexible size and strength mechanism. If this fails, they will need to seriously consider getting help, reducing other attributes or simplifying the mechanism to stay on track.

	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5
Prototype Goal	• Develop brochure of target coffee maker	• Develop 3D video highlighting flexibiliy coffee making	• Fully working prototype	• Production-ready prototype	• Production Starts

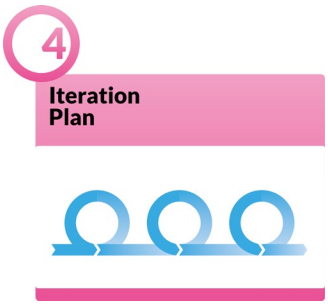
Major Deliverables	• Preliminary concepts
	• Interface technology decided
		• Preliminary design
		  • Electronics
		  • Smart features
		  • Industrial design
			• Integrated device
			• 80% of features in prototype
				• Requirements lockdown
				• Mechanical design done
				• Electronics done
					• Production
					• Manual
					• Website
					• Materials
					• Packaging

Customer Feedback Gaols	• Test value proposition	• Test value and voice control initial implementation	• Test flexible size and strength design	• Test for production readiness	• Marketing messaging and launch plan
Other Goals	• Get momentum with team	• Cost estimates	• Order prototype molds	• BOM complete • Launch plan • Final molds • Certification	• Sales ready • Channel ready • Pricing ready

ITERATION PLANNING: Agile Project Plan Checklist

OVERVIEW

Iteration planning (both the first time as well as before upcoming iterations) is also a good time to review many of the elements associated with HW and cross-discipline efforts. The table below shows a range of these elements and acts as a checklist for the team to make sure they have been considered each element and how it fits into the overall iteration plan.



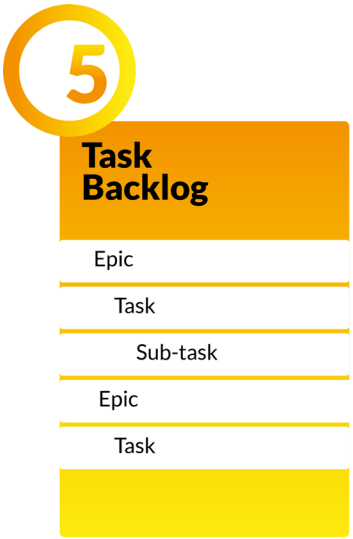
For example, the team knows they will need to consider patents and intellectual property somewhere in the plan, so they determine which iteration the activities will start. During upcoming iteration and sprint planning sessions, they will break this category down into tasks and sub-tasks.

Needed?	Project Plan Element	Owner
Yes	Target customer and needs identified and personas	Jim
Yes	Assumptions, constraints, risks and major dependencies	All
Yes	Product/system architecture and major attributes	Frieda
Yes	Project staffing and cross-functional project support	Juan
Yes	Roles and responsibilities (including external stakeholders)	Juan
Yes	Financial plan, forecasts and budget	Jim
Yes	Capital expenses, equipment and tools	Jim
Yes	Project metrics and KPI's	Juan
Yes	Documentation needs	Jim
No	Training plan	
Yes	Patents and intellectual property	Chester
Yes	Standards and certifications	Carol
Yes	Design for manufacturability, servicability, etc.	Lehi
Yes	Testing/validation/QA plan	Lehi
Yes	Release/Launch plan	Jim

Developing a MAHD Task Backlog

IT'S TIME TO DEVELOP THE LIST OF TASKS FOR EXECUTION

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 a few days or 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 many details will be 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. For example, a user story will not typically be a specific backlog item. Generally the MAHD Task Backlog is a prioritized list of development tasks that are related to user stories and product attributes, but not these items themselves.

To refer back to our coffee maker example, the backlog might include:

1. Investigate flexible strength technology options and determine the best approach.
2. Investigate smart devices and determine the priority of which ones to support.
3. Develop a range of industrial design concepts for the look and feel of the coffee maker.
4. Develop the user interface electronics subsystem.
5. 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 fourth one, "Develop the user interface electronics subsystem..." 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:

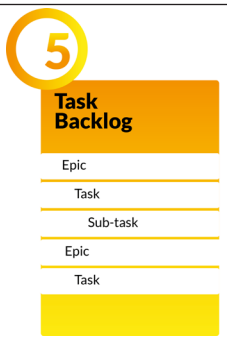
- Define the number of buttons needed and their functionality.
- Develop preliminary interface board for prototype 2.
- Etc.

Once your MAHD Task Backlog is ready you can then begin planning your first sprint and you're off and running with Agile. Throughout the Agile process you'll continue to clarify and prioritize backlog tasks, refine estimates, move specfic tasks into sprints and track progress.

TASK BACKLOG: Step-by-Step

DESCRIPTION:

The Task Backlog becomes the set of execution tasks for the development team. Each item is prioritized based on the overall iteration plan that has already considered dependencies, prototype needs, risks and customer importance. As sprints are planned, developers commit to completing each task. Work is not complete on the task until the acceptance criteria are met.



STEP-BY-STEP:

1. Each cross-discipline team reviews the iteration plan and develops a list of tasks related to their area. This may include industrial design, electronics, mechanical, production, etc.
2. Regroup as a team (the whole team or project leads) and review the tasks. As a team, decide how to organize the tasks and determine which tasks have priority.
3. Major tasks (epics) are broken down into smaller tasks in preparation for sprint planning.
4. The teams review the results and clarify the tasks and ensure they match the high priority user stories and the iteration plan.
5. The task backlog is updated by the APM and dev leaders before each sprint and iteration.

WHAT CAN GO WRONG?

1. If only one person develops the backlog, there will likely be little support for tasks. This will lead to long and difficult sprint planning sessions.
2. The backlog only includes items for one or two sprints. The backlog should represent the whole project — at least a high level.
3. As detailed engineering tasks begin to fill the backlog, it's easy to lose focus on the big picture.

TIPS FOR SUCCESS:

1. Ensure priorities and tasks are clear for at least a whole product iteration at a time. This allows team members to quickly move to the next tasks.
2. Backlog items don't need to be in the form of "user stories." This is OK for SW, but as noted earlier, it doesn't usually make sense for HW.
3. Ensure the tasks are related to developing attributes that support important user stories.

WHO OWNS THIS ACTIVITY?

1. **Dev Team Leaders** - This activity must be completed as a cross-discipline team, guided by the APM.

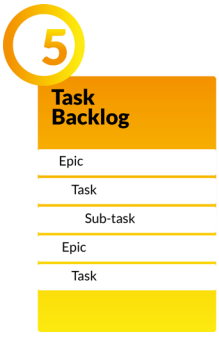
WHEN DO YOU DO IT?

Required to initiate an agile project and updated before each sprint and iteration planning session.

TASK BACKLOG: Template and Example

OVERVIEW

The table below shows a sample of a task backlog for the coffee maker. Using the iteration plan as their guide, the team groups each task into categories of design, subsystem or functionality. Tasks are assigned priorities to ensure high importance items are addressed early, dependencies are considered and all the major deliverables are taken into account. Large tasks are labeled as epics and are further broken down into sub-tasks. An example is shown for the epic "Flexible strength control." This was a highly desired user story and has both technology and customer acceptance risk, so these are high priority tasks.



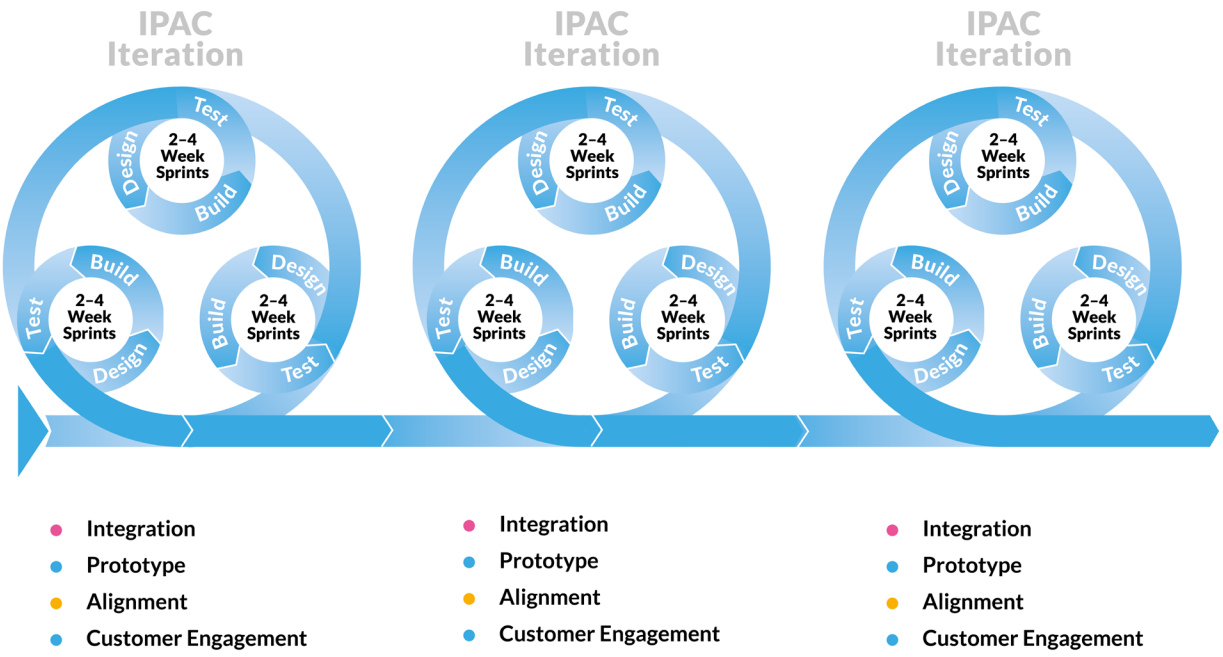
This team uses Agile management software to document the backlog in preparation for the first iteration and sprint planning session, but using a whiteboard is OK.

Category	Epic	Task	Pty.
Physical Interface	Interface technology	Task 1...	1
	Programming buttons	Task 1...	2
	On/off mechanism	Task 1...	3
	Temperature control	Task 1...	2
Physical Design	Flexible strength control	Investigate possible technology	1
		Develop strength testing mechanism	1
		Develop criteria and agree on approach	1
	Color choices	Task 1...	2
	Materials selection	Task 1...	2
	Coffee grinder mechanism	Task 1...	1
	Water access mechansm	Task 1...	2
	Flexible Size control	Task 1...	1
	Design concepts	Task 1...	1
	Filter method	Task 1...	3
	Thermal Carafe Design	Task 1...	2
Smart Features	Voice control	Research smart device protocols	1
		Select initial device to support	1
	Smart device interface	Task 1...	1
Other Tasks	Prototype 1 - Brochure	Develop brochure	1
	Prototype 2 - 3D model/video	Task 1...	2
	Prototype 3 - Full working	Task 1...	3
	Organize customer panel	Task 1...	1
	Etc...	Etc...	

Planning MAHD Sprints: Execution Cycles

KEEPING THE AGILE ENGINE FINE-TUNED

When most people think of "Agile," they imagine the short development cycles where small teams are committed to delivering a set of features selected from a backlog. This provides the foundation of Agile methodology. Executing MAHD sprints is similar. The only difference is that hardware backlog tasks 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 vendors, 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.



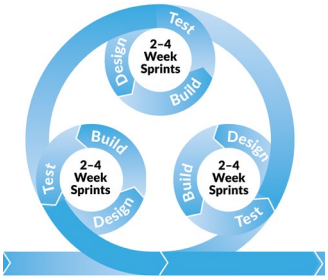
When planning for sprints, MAHD cross-discipline teams will review the task backlog, clarify tasks and determine the number of tasks they can commit to during a two-to-eight week sprint. As with any Agile process, the governance process is based on regular meetings where teams share their progress, groom tasks for the next sprint and validate their work. Teams may 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.

SPRINT PLANNING: Step-by-Step

DESCRIPTION:

Planning the tasks that get executed each sprint is one of the most important elements of Agile. It's the time to review progress, select tasks from the backlog and ensure the team is on track. While Agile for SW often uses 1 to 2 week sprints, HW sprints are typically longer and last from 2 to 4 weeks.



STEP-BY-STEP:

1. Plan sprints the week prior to a new sprint starting so the team is ready to go as soon as the current sprint is complete.
2. Before sprint planning, the backlog should be updated with the latest thinking and tasks.
3. Project leaders from each function should be clear on the tasks and prepare for sprint planning including adding acceptance criteria to each task.
4. As tasks are selected, use estimation methods such as planning poker (described later) to estimate the size of each task. Tasks larger than one sprint are broken down into sub-tasks.
5. As tasks are completed, get agreement from stakeholders that task results are acceptable.

WHAT CAN GO WRONG?

1. Not preparing for the sprint meeting leads to long planning sessions and frustration.
2. It's OK to carry a task over to a new sprint, but if it is common, then something is wrong with estimation, planning dependencies or other factors.
3. Not removing roadblocks during a sprint before sprint planning (or at least having a plan).
4. Letting external dependencies slow the team down.

TIPS FOR SUCCESS:

1. Team members must commit to their own tasks. No one should commit for them.
2. A commitment is a contract with the team. "I will get this done!"
3. Always have high priority items groomed for the next sprint in case team members have time or are stuck with their current tasks.
4. For iterations, do the same activities, but at a higher level, to ensure the overall plan is on track.

WHO OWNS THIS ACTIVITY?

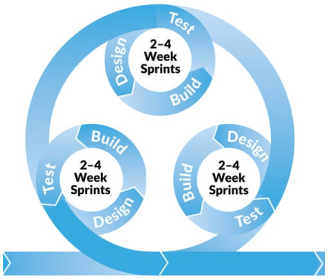
1. The **APM** - While this activity must be completed as a cross-discipline team, the APM facilitates success.

WHEN DO YOU DO IT?

One week before the beginning of the next sprint. This is in parallel with closing out the current sprint.

SPRINT PLANNING: Template and Example

OVERVIEW



As the coffee maker team prepares to execute, they conduct their first sprint planning session. With the initial iteration in mind, including the paper (brochure) prototype and design milestones, each project lead and team member selects tasks from the backlog. For some tasks, they let the team member estimate the duration. For other more complicated or less understood tasks, each team member estimates the task and they come to a quick concensus on how long it should take. Larger tasks are broken down into smaller tasks to fit within the sprint. Since this is their first sprint of the iteration, they also roughly define the sprints for the whole iteration to ensure they can achive the milestones.

Once everyone agrees on the sprint tasks, they are ready to execute.

Iteration 1, Sprint 1	Task	Pty.	Owner	Acceptance
Physical Interface	Task 1...	1	Linda	Internal review
	Task 1...	2		
	Task 1...	3		
	Task 1...	2		
Physical Design	Technology for flexible strength	1	Joschua	Internal review
	Design strength testing procedure	1	Gerri	Internal review
	Develop criteria and agree on approach	1		
	Task 1...	2		
	Task 1...	2		
	Task 1...	1		
	Task 1...	2		
	Task 1...	1		
	Task 1...	1		
	Task 1...	3		
	Task 1...	2		
Smart Features	Research smart device protocols	1	Ted	Internal review
	Select initial device	1		
	Task 1...	1		
Other Tasks	Develop brochure	1	Jim	Marketing review
	Task 1...	2		
	Task 1...	3		
	Task 1...	1		
	Etc...			

SPRINT PLANNING: Estimation Techniques and Tips

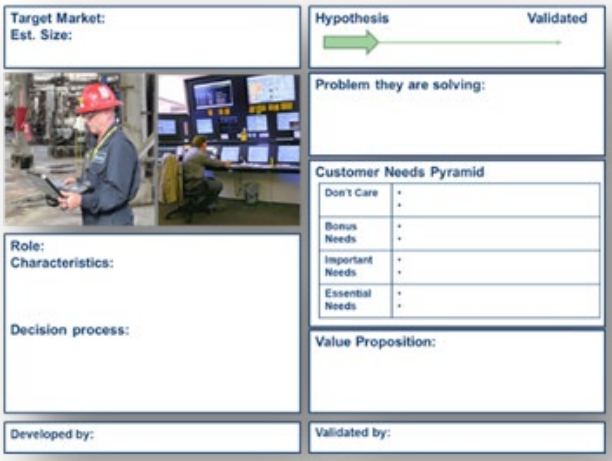
One of the most challenging activities, as well as one of the most important, is the process of grooming tasks where they are clearly defined, broken down and their duration estimated. There is no perfect estimation technique. Many SW Agile teams use a point system where a point is loosely associated with a unit of time. MAHD practitioners can use a similar approach, but due to the cross-discipline nature of projects and wide range of activites, points don't always make sense. Many teams opt to use person-days (E.g. six person days is one person for six days or two people for three days.)

The following table offers some suggestions for estimating the duration of tasks, iterations and whole projects. What's most important is to use an estimation technique consistently so you can track progress and improve over time.

Purpose	Technique	Description	Notes
Whole project initial estimate	Iterations and Sprints	During MAHD On-ramp planning, consider how many iterations the whole project will take. Then estimate the number of sprints and their duration. Multiply the total number of sprints by their duration to get a rough project estimate.	This works best with an experienced team after a couple of Agile projects have been completed.
	Task rollup	Once the whole task backlog has been created, estimate the duration of each major task as possible. Add all of the task durations and include a buffer for missed tasks and inevitable task breakdowns.	This is a good way for Agile teams just starting since it's similar to traditional project management.
	Analogy	Use past projects as a guide for the current project. If a similar project was 12 months, but this one is more complex, identify the deltas and estimate.	This can work in conjunction with any technique as a point of comparison.
Task estimates	Planning poker/ T-shirt sizes	Use points to estimate small, medium, large, extra-large, etc. tasks. Agree that a "medium" can fit within one sprint, etc. Planning poker often uses a Fibonacci sequence of 0, 1, 2, 3, 5, 8, 13, 21 or similar to represent sizes.	A faster process, but less granular. This is generally how Agile for SW does estimation.
	Delphi	Let the experts decide. For each functional area, allow anyone familiar with the task to offer an estimate. Discuss and average them or agree on the best logic.	This works well in combination with "Task rollup" shown above. This is typically the default method.

ENGAGING CUSTOMERS: The Fuel for Success

While successfully managing Agile projects using the steps and tools already discussed is important, you won't be truly agile until you build in customer feedback loops. This set of activities is often difficult for teams since it requires new skills to identify the right customers, interview customers in ways that reveal true insight and integrate customer feedback into development efforts. Engaging customers also requires time and resources that many teams feel can be better spent on development. However, only by engaging with real customers at the right points in your Agile process will you have the guidance necessary to make tough decisions and keep the development team on track.



Customer Personas Describe Target Customers

Some of the elements required for getting customer insight at the right times include:

1. **Develop a clear understanding of the customer** - You'll need to identify decision-makers, users and influencers and determine which type of customer is most important to understand and what type of feedback you need. Using customer personas can help the team understand who the target customer is and what's important to them.
2. **Determine how you will find and engage with customers** - You'll need to find representative customers that want to spend time with you and provide meaningful insight. Customers can't be fellow developers, other company employees or management.
3. **Build interviewing and listening skills** - You'll need to learn how to engage with customers using good interview skills to go deep into attitudes, needs and reactions to your solutions.
4. **Aggregation and integration of feedback** - You'll need to clearly summarize and priortize what you hear from customers and confidently incorporate your learnings into backlog items, feature refinement and decision-making.

One tool that teams have found invaluable in order to gain fast, accurate customer insight is a customer panel. This is a set of target customers that have been asked to participate in the company's success, have an interest in the product under development and are able to articulate their needs and concerns. The panel is maintained by the APM or a designated person and are called upon for meetings, phone interviews or just to answer quick questions when pressing decisions need to be made.

Getting Started with MAHD

We know we've only touched the surface on 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 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.	2-4 week sprints grouped into whole product iterations to test key features and specifications.
Requirements	Defined by user stories.	Defined by features, design targets, user stories, and business needs
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 and often several cross-discipline leaders.
Process Owner	Scrum Master	Agile Project Manager
Team Orientation	SW Teams	Cross-discipline Teams
Focus Matrix	Not used	A crucial planning activity 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

Or visit Auxilium and browse our services at:

www.Auxilium-inc.com

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

Auxilium