Hardware Agility with SAFe's Principles
Duane Bushman

SPCT, Sr Enterprise Transformation Coach
Rockwell Automation
Icon Agility Services
Can hardware be done with agility?

- Is it possible to build a detailed project plan that accurately captures a complete hardware development cycle prior to development?
- Do detailed hardware project plans remain stable over time?
- Is it possible to build a concise set of hardware product requirements that will satisfy customer and market need 6, 12 or 18 months in the future?
- Do hardware product requirements remain stable over time?
- Are hardware products consistently delivered on schedule?
- Do hardware products consistently meet market demand and customer need?
- Are target technologies thoroughly understood and unlikely to change?
- Would a hardware team design and build a product the same way twice?
- Are product architectures likely to remain stable over time?
- Are people and their respective skill-sets interchangeable?

If you answered 'No' to most of these questions, you should consider switching to agility.

As in software development, hardware benefits from lean-agile approaches.
Yes, Hardware is Different – but SAFe Principles still apply
Consider some Hardware Differences…

1. Lead time is longer & cost is greater
2. Cannot deliver a “partial” product
3. Specialized skills and non-fungible roles

What SAFe Principles Apply?
Hardware lead time is longer & cost is greater

- **Procurement** takes time and **limits** number of design iterations
- Hard to recover from **late changes**
- Changes to regulatory standards or obsolescence of Hardware components are **disruptive**
- Architecture and high-level implementation need to be **locked down** very early in design

What **SAFe** **Principles** **Apply?**

#1-Take an economic view
#2-Apply systems thinking
#3-Focus on visibility and commitments
#4-Build incrementally with short, fixed-length learning cycles
#5-Base milestone criteria on value delivery, not working systems
#6-Visualize and limit WIP, reduce batch sizes, and manage queue lengths
#7-Apply cadence, standardize schedule-domain planning
#8-Unlock the intrinsic motivation of knowledge workers
#9-Decentralize decision-making
1. Understand **design trade off** decisions & make them quick
   - **Avoid** decision made & reversed or lack of a decision
   - Push quality left; iterate.
   - Faster feedback loops decrease cost & risk
   - Incremental delivery enables **informed** decision on when to shift to “real” HW and get it out the door

#1 – Take an economic view
   - #2-Apply systems thinking

#3 – Assume variability; preserve options
   - #4-Build incrementally with fast, integrated learning cycles
   - #5-Base milestones on objective evaluation of working systems
   - #6-Visualize and limit WIP, reduce batch sizes, and manage queue lengths
   - #7-Apply cadence, synchronize with cross-domain planning
   - #8-Unlock the intrinsic motivation of knowledge workers
   - #9-Decentralize decision-making

Lead Time is **longer**! Added discovery contributes to; changes, procurement, tooling; we must preserve options
   - We **cannot** know everything on day 1
   - Fixed vs. Variable Solution Intent
   - **Maximize** the Plan Do Check Adjust (PDCA), Define Build Test (DBT)…
   - Use **Models** - what questions do we want this model to answer?
Cannot deliver a “partial” product

- HW delivery **cannot** be broken down into features and individually / independently developed
- Qualification & regulatory testing require **entire** product to be functional
- Hardware systems are **complex** with interaction between subsystems
- **Partial** products cannot be tested or delivered to the customer
- We cannot deliver a partial product in months much less in **two weeks**!
Cannot deliver a “partial” product

- **Significant #** of integration pts & communication channels
- Agility beyond Dev to Ops, Indus, part procurement, proto build scheduling…(Incremental vs BOMs “finalized” early)

- **We need to **reframe** the conversation.
  - What can we deliver/demonstrate that a stakeholder cares about in two weeks?
  - What can we **learn**?
  - What can we **integrate**?
  - How can we **retire** risk?
  - What will **validate** we are on the right path?
  - The answers to these types of questions become the definition of done for an iteration.

- Use **Definition of Done** for design maturity

**Construction** Milestones vs. Customer Deliverables

- Cross-functional design reviews
- Systems portable? Video (live(recorded)), models, computer generated, augmented reality…
Specialized skills and non-fungible roles

- We have a large # of roles needed & functions are not fungible
- Infrastructure resources (laboratory, equipment) are very expensive and cannot be replicated to support multiple development projects simultaneously
- This includes development laboratory, equipment, as well as manufacturing
- People with very specialized skills can provide support mainly in a serial fashion
Specialized skills and non-fungible roles

- Whether SW, FW or HW, top challenge is **Too Much WIP**!
- All this WIP is vying for the same skilled people & resources

- Cadence – with high # of contributors, a “healthy heartbeat” is key to alignment and managing integration points
- Synchronization – make all work, risks & impediments highly visible
- Cross-functional ART pulls disparate groups together to plan
- Key is **dependency management** and ability to **track & maintain**. Both are important operationally beyond PI Planning

Establish the **guardrails**, get out of the way
- Teams own and solve their own problems
- Teams empowered to seek out and **eliminate non-value** added work
- Balances **intentional** architecture and **emergent** design

#6 – Visualize and limit WIP, reduce batch sizes, manage queue lengths

#7 – Apply cadence, synchronize with cross-domain planning

#8 – Unlock the intrinsic motivation of knowledge workers

#9 – Decentralize decision-making
Horizon “Roadmap”
Find the Roadmap Balance

Consider using a “Horizon Roadmap”

- Fills the gap and provides visibility beyond 3 PI’s
- Contains development milestones supporting the long “lead time”
- Socialize HW “Big Rocks” where Procurement or Production Readiness Pace Key HW Development & Manufacturing Milestones
Development milestones support long lead-time items like securing parts, board builds, lab time and test certifications...

Solution Roadmap – Epics (18 – 36 months)

Horizon Roadmap

PI Roadmap – Features (3 PI’s)
Roles – ART & Team Composition
Potential communication channels is \( n \frac{n - 1}{2} \)

Generic ART has 9 functions that is 36 communication channels
Expanded Functions on the ART

• HW has significant # of **skillsets** involved to produce the final product.

• Means significant # of **integration points** and **communication channels**.

• What if you took one person from each **function** that contributes to the value delivery in hardware?

17 expanded functions is **136** communication channels
Agile Team – All Roles Still Apply

• **Cross-functional** Development Team is still the goal
  
  • SW Dev Team may contain Dev, Test, Biz Analyst…
  
  • HW Dev team may contain:
    
    • Engineering: Power, Mechanical, Electrical…
    
    • Industrialization
    
    • Hardware Tech & Test
    
    • Manufacturing Test
    
    • *All* the skills needed to deliver value
Skillsets Change during Course of HW Development

- As Design **matures**, skillsets **change**
  - Early on Design: **significant** Power Eng, **partial** Mech Eng, **minimal** HW Tech & Test
  - Dev & Execute: **significant** HW Tech & Test, **minimal** Power Eng, Mech stays consistent

- Reality of **non-fungible** roles but enhances **alignment**

- **Supports** long-lived and cross-functional teams
Agile Team – Yes – All Roles!

- What about **SM** and **PO**?
  - **Absolute** Must Have!
    - Don’t “assign” SM or PO role to team members
    - Engineers want to be engineers!
  - **SM** - Coaches & Mentors team
    - Addresses the mechanics of being an “agile” team
    - Helps account for **Cultural Change**
  - **PO** - Technical or Business? Does it matter?
    - Isn’t the PO the **proxy** for the customer?
    - Our work is very technical
    - Consideration / trade offs of each **approach**
PI Planning and “Incremental Delivery”
There is no magic in SAFe
… except maybe for PI Planning
What we need in PI Planning

- **Socialized** Roadmaps
  - Solution, PI and YES, a *Horizon* Roadmap
  - Aids in addressing the long lead time and increased cost

- **Well-formed** Enterprise Backlog
  - Alignment via Enablers and Capacity Allocation
  - Cross-functional deliverables

- Align to a PI **Theme**
  - Entire ART with common goal
  - Incremental delivery and investment decisions

This is no different for ANY PI Planning!
What is Different for PI Planning with HW?

Remember the expanded functions for 136 communication channels

- More **channels** means more **contributors**
- These functions need to be **present** including Operations & Shared Services
- An Invite is a great start, there is **more** to do
  - Do they understand their role, ceremonies, expectations?
  - Did you account for working space and supplies?
  - What about on the Program/Dependency board? Will require swim lanes or BVIR
  - Are all epic owners present including **Enabler** epic owners?
  - What **Enabler** Capabilities do we need to align across ARTs?
  - Do we understand the priority of the **Enablers** requested?
  - Who is going to **coach** the teams on the enablers?
What about testing and system integration?
Our Goals are the Same - Test Early and Often

- **Environment** management
  - Costs more for "environments" than software
  - Costs for **physical** set up and test **targets** (*# boards you will destroy*)
  - Outside test assets
  - Outside **compliance** bodies such as UL, TUV
  - Physical assets - Work cells, dyne (powertrain test solution for motors)
    - Environmental Chambers; Thermal Chambers; Shock and vibe;
    - Wind and water (consider hurricane strength)…

- How often should we **integrate**?
  - Must have a **base** product
  - Integrate too early without a base; we waste **learning** cycles
  - Integrate too late; changes significantly impact lead time; **procurement**

- Use **Models & Simulators**
  - Hardware-in-the-loop (HIL), Flying Probe
  - Drawing, sketches, breadboard, brass board, mathematical, 3D, 2D, throw away, scale, form, study, functional prototype, user interface prototype, low fidelity, mockup, proof of concept, simulations, storyboard, wireframes, etc.
Final Thoughts
While there are differences, there are more similarities

- Get out of your own way…agility applies, embrace it
- SAFe applies! The **Principles** are the foundation
  - When challenged or doubting the path, ask “What principle applies and how can I apply it?”
- Hardware functions are extensive
  - Define and train all functions together
  - Non-traditional HW roles like SMs and POs are critical and foster the new mindset
  - Roles filled in the right ratios, and it is OK if their backgrounds vary!
- Develop your “**Horizon Roadmap**”. Balance early indicators with longer visibility.
- Cross-functional teams work!
  - Leadership must understand and supports the “new” team structures.
  - Scarce & Shared Resources are a challenge – acknowledge, embrace, iterate and experiment
- Pull it altogether in **PI Planning**!
- Embrace SAFe's body of knowledge and **proven practices** to continuously improve, learn, and deliver value to our organizations.
Questions
Special Thanks

Peer review and contributors:
Bob Kulinski – RGK Solutions
John Miller, SPCT – Icon Agility Services
Thank you!

Coming soon – presentation downloads at global.safesummit.com/presentations
Please rate sessions

1. Click the Schedule icon in mobile app and locate the session.

2. ‘Check in’ by clicking the plus sign next to the session title.

3. Tap star rating at top of screen.