SYNAPSE

A Cambridge Consultants Company

MASTERING HARDWARE PRODUCT DEVELOPMENT

08

Levers for Optimizing Your Product's Cost, Schedule, and Quality

By Jeff Hebert, President, Synapse Product Development

CONTENTS.

WHAT WILL IT TAKE TO BUILD THIS HARDWARE PRODUCT?

“How long will it take?”

“How much will it cost?”

These are the first two questions most prospective Synapse clients ask when they approach us to develop their hardware product concept. The conversation often evolves to “what is possible within this budget and timeframe?” We understand it’s nearly impossible to escape answering these questions as part of the business plan to justify a significant product development investment.

A challenge for Synapse in answering accurately is that most of the products we help clients create are innovative—the first of their kind. In novel development programs, the lines between research and development are blurred and it’s likely that both the concept and the solution will undergo significant iteration along the way. Needless to say, providing good estimates of cost and duration up front is not easy. Underestimation risks leaving clients without enough budget to realize their ambition. Overestimation risks clients perceiving us as conservative or expensive.

To better estimate for clients, we’ve done what any engineer would do—we built a case study database and ran some numbers! Our goal in doing so was to understand the variables impacting development scope so we could provide our clients with not only more accurate estimates, but also an understanding of the levers they can utilize to work within target budgets and schedules. Here are the categories that emerged, based on our nearly 20-year history of hardware development consulting for hundreds of clients and products:

01 **Clarity of the Product Purpose**

02 **Maturity of the Underlying Technology (TRL)**

03 **Magnitude and Complexity of the Product Ecosystem**

04 **Quality, Performance, and Compliance Requirements**

05 **Product Cost Optimization**

06 **Build and Launch Volumes**

07 **Development Budget and Schedule Constraints**

08 **Risk Tolerance**

The rest of this ebook walks through each category, explains how we approach and assess it, and provides examples from our experience. We hope this knowledge will help veteran product companies and hardware startups alike.

< Contents

01

CLARITY OF THE PRODUCT PURPOSE

WHY.WHAT

HOW

Nearly every Synapse client does not begin our engagement with a clear Market Requirements Document (MRD) and Product Requirements Document (PRD). This is ok! In fact, while definition and clarity are helpful, we recommend focusing on the “Why” and the “What” in your definition, and leaving the “How” open at first. This will give the development team enough direction to work efficiently and reduce churn while keeping the solution space flexible, avoiding pigeonholing with an approach that might not be the most efficient.

For example,

Propeller Health wanted their inhaler to log the location of the user whenever the device was used. This initially led them down the path of prescribing on-board GPS as a requirement, which would have been very expensive and power-hungry. Through further iteration of the product definition, it became clear that there were other acceptable ways to access the critical location data. The ultimate product instead used Bluetooth® connectivity from the device to a mobile phone to use the phone’s GPS, saving a significant amount of effort, time, and product cost.

Image Credit: Propeller Health, propellerhealth.com

02

MATURITY OF THE UNDERLYING TECHNOLOGY

Building a product with off-the-shelf (OTS) components and technologies already proven in the market removes significant risk from a program and increases the chances that fewer design-build-test iterations will be required

OTS

(the number of iterations in later stages of development is the biggest killer for hardware development schedule and budget). Going completely OTS is the fastest way to market and can enable a first-mover advantage, but also means the barrier to entry for competitors is lower.

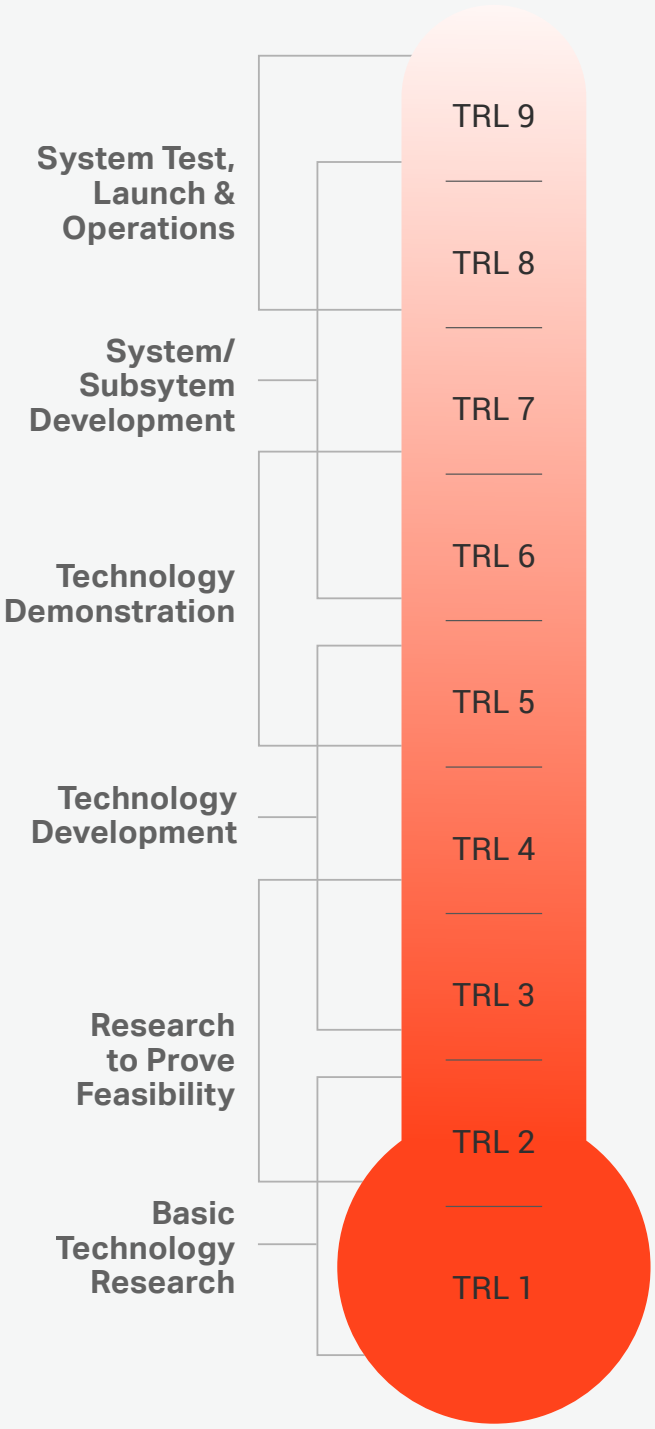
Many projects require new, unproven technologies. While these projects result in bigger breakthroughs and more defensible competitive differentiation for clients, they inherently involve more risk and can take much more time and energy to develop. This means we need a way to assess the risk and appropriately balance the level of invention with speed and cost

To assess technology maturity, we look to the **Technology Readiness Level (TRL) system created by NASA**. The 9-level scale ranges from modeling and lab-based experimentation, through subsystem demonstration, to system prototypes proven in the operating environment, and on to launch and successful operation in the market.

Considering subsystem and component maturity in this way helps highlight risks and the efforts that will be needed to mitigate and address them in order to progress to higher levels of maturity.

For example, if the core product we're building requires development of a novel sensor, the development effort and schedule if we're starting at TRL 5 is significantly less than if we're starting at TRL 3. Starting at TRL 3, we'd need to create a physical prototype of the sensor subsystem and test it in the lab, then prove it in a relevant operating environment to match TRL 5—easily weeks, if not months of work.

While fantastic for assessing underlying technology, a clear shortcoming of the NASA TRL system for high-volume hardware development is that it doesn't consider mass-production readiness. We also need to know that the supply chain and production systems are capable of delivering the volumes needed in the timeframe desired. Our New Product Introduction team focuses on this assessment, starting early in the development lifecycle, as it can significantly impact design strategy.



By NASA/Airspace Systems (AS) - http://as.nasa.gov/aboutus/trl-introduction.html

< Contents

03

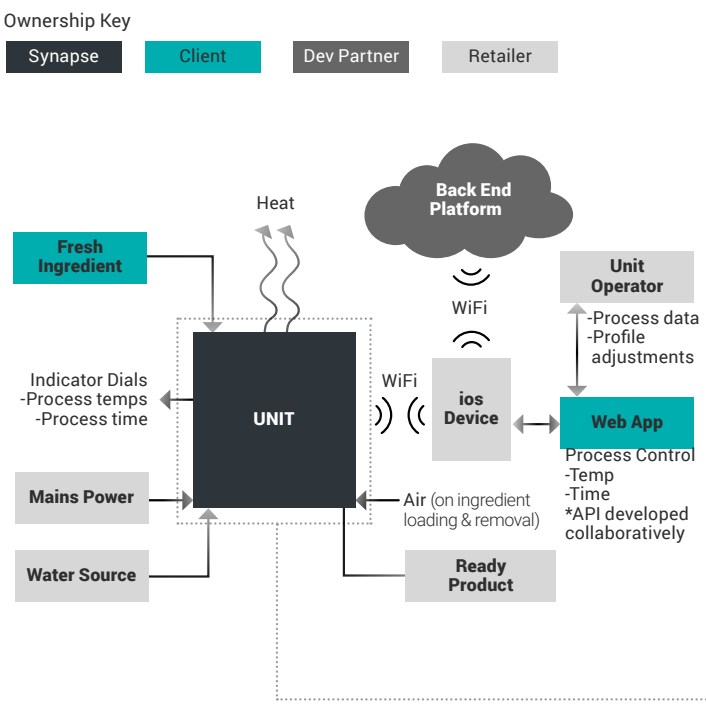
MAGNITUDE & COMPLEXITY OF THE PRODUCT ECOSYSTEM

One of the key reasons to flesh out the “Why” and the “What” of a product early is to enable drawing a full product ecosystem diagram. This exercise often identifies aspects of a product which need to be considered and developed that might not have been part of the original vision or budget expectation. It’s a really easy step to skip, assuming it’s obvious to everyone involved, but it always illuminates gaps and helps teams get on the same page.

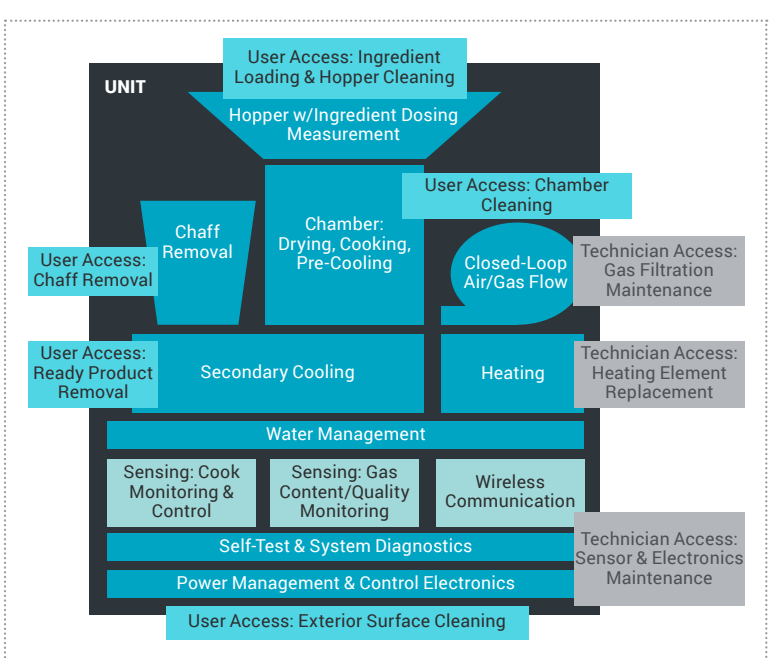
Creating an ecosystem diagram highlights physical system elements (like charging docks and adapters), software ecosystem elements (like cloud storage, algorithms, and mobile user interfaces), secondary product “users” (like RMA technicians, customer support, and retail staff), interfaces (like wireless protocols, buttons, audio, charging ports), and even logistics (like supply chain and user-facing packaging). Many of the things I’ve just listed are common sources of pitfalls in complex hardware product development. Realizing their existence and importance late in a development program can be very costly.

The system diagram is an easy way to assess how complex the ecosystem is, which directly drives the scale of the development program.

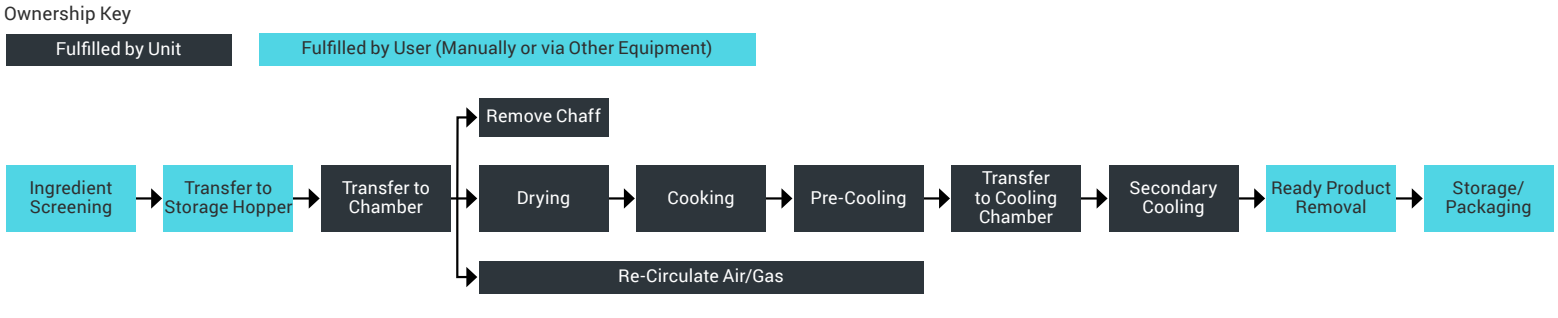
Product Ecosystem Diagram



High-Level Functional Diagram

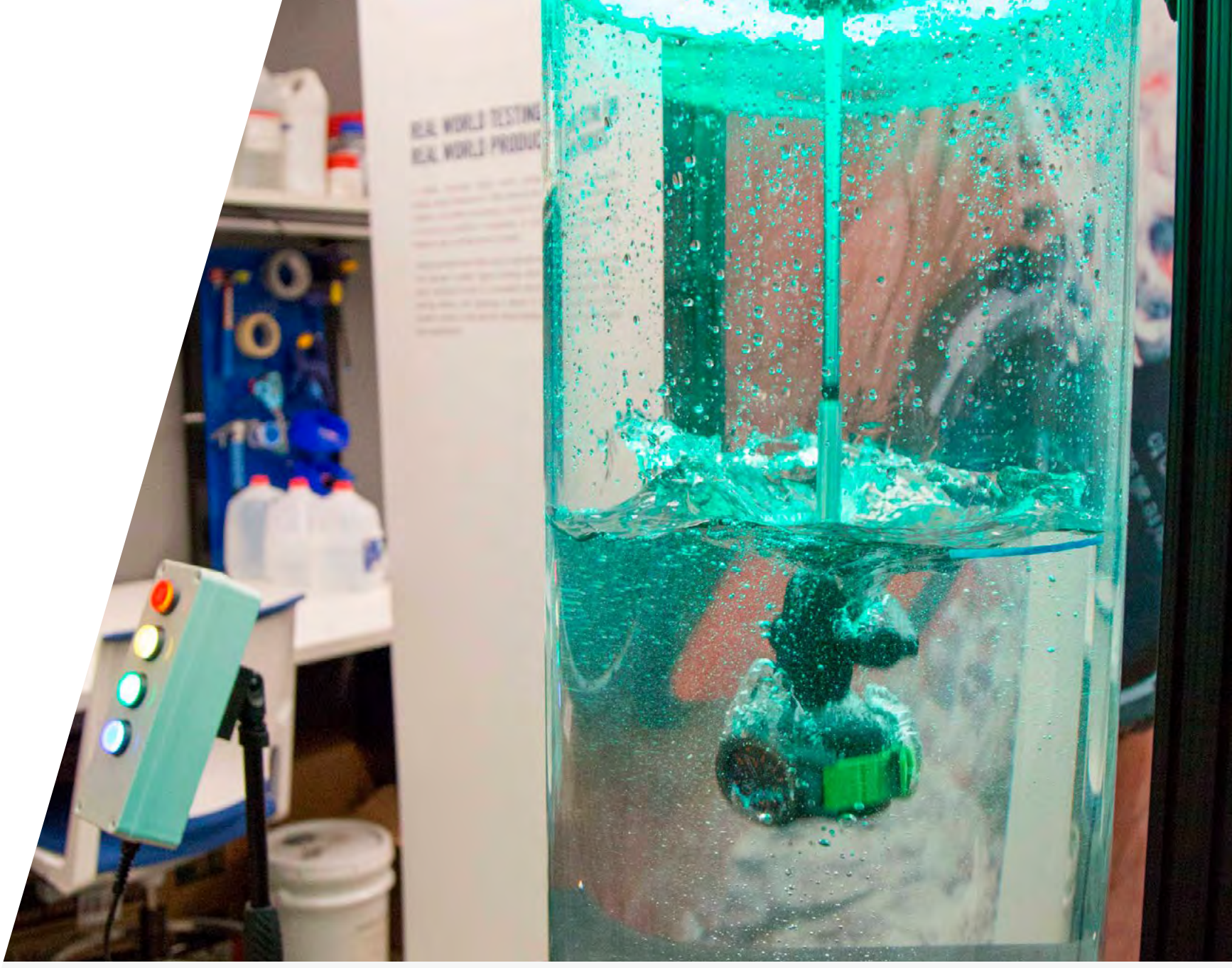


High-Level Process Flow



04

QUALITY, PERFORMANCE, & COMPLIANCE REQUIREMENTS



While some of these aspects are fuzzy and difficult to pin down, they can represent make-or-break aspects of the product value proposition for customers and can also drive the development budget and schedule significantly. Everyone wants a classic Apple level of polish until they see the price tag!

Included in this category are a number of aspects, some of which are obvious and some of which are easy to forget.

- **The breadth of use cases and functionality**
- **Performance KPI aggressiveness**
- **Extremeness of the operating environment**
- **Compliance and certification needs**
- **Expectations of reliability and useful life**
- **Form factor aggressiveness**
- **Cosmetic expectations**

For example,

the Nike FuelBand had incredibly strict and aggressive form-factor and cosmetic requirements—a hair thicker and it might have been canceled. These requirements ended up being essential to drive consumer adoption and success of the iconic, first-of-its-kind product, but they also led to many inventive engineering solutions and iterations during the product development process. The novel solutions implemented, such as the first-ever curved rechargeable batteries and a directly-overmolded FPCA and LED display array, took a lot of time and energy to create compared to a standard smartwatch design with all of the electronics housed underneath a standard display.

PRODUCT COST OPTIMIZATION

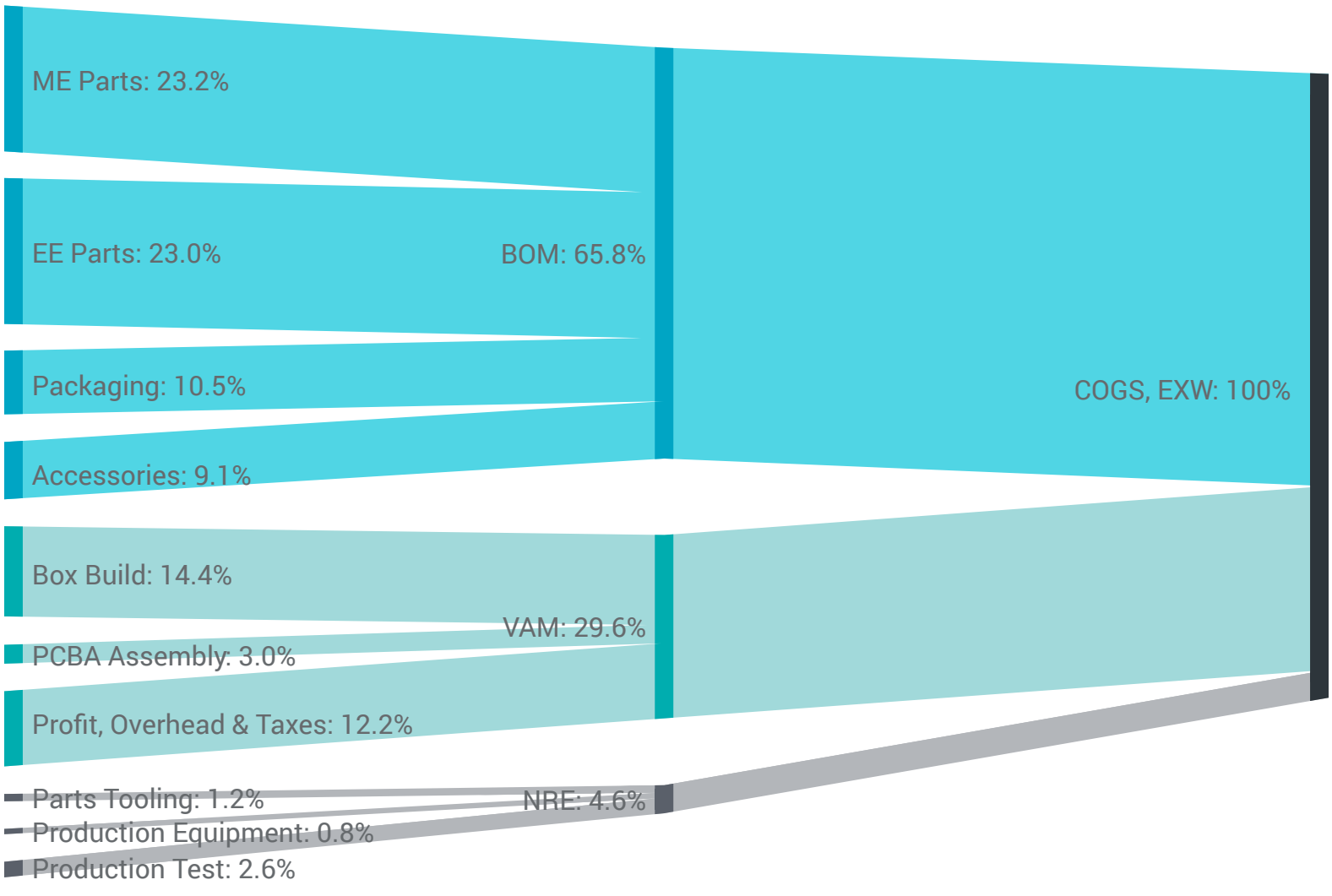
Not to be confused with cost of development,

here we consider how much time and energy should be applied to Design for X (DfX), New Product Introduction (NPI), and supply chain optimization to keep unit costs low. Product cost is a critical part of the business plan and depends heavily on margin targets and expected sales price elasticity.

The product cost target permeates the entire design, from key component selection to assembly procedures to test methods. An aggressive cost target with the same desired use cases and feature set will take more engineering and strategy effort to achieve. The higher the sales volume, the more easily effort in cost optimization will pay for itself over time.

If you're looking to release a first-generation product to market quickly and in lower volume, it may make sense to put minimal effort into this area and absorb a higher product cost until cost-down efforts can be justified. If you're releasing a high-volume product with thin margins, cost optimization can be a make-or-break activity.

Below is a visualization we use to consider the full cost of goods sold (COGS) for a product. Many engineering and product development teams focus exclusively on the bill of materials (BOM), but transformational costs and prorated non-recurring expenses can be significant factors. Looking at the full picture and prioritizing areas of focus based on scale and ability to impact them is a very worthwhile exercise to start early and come back to throughout the project.



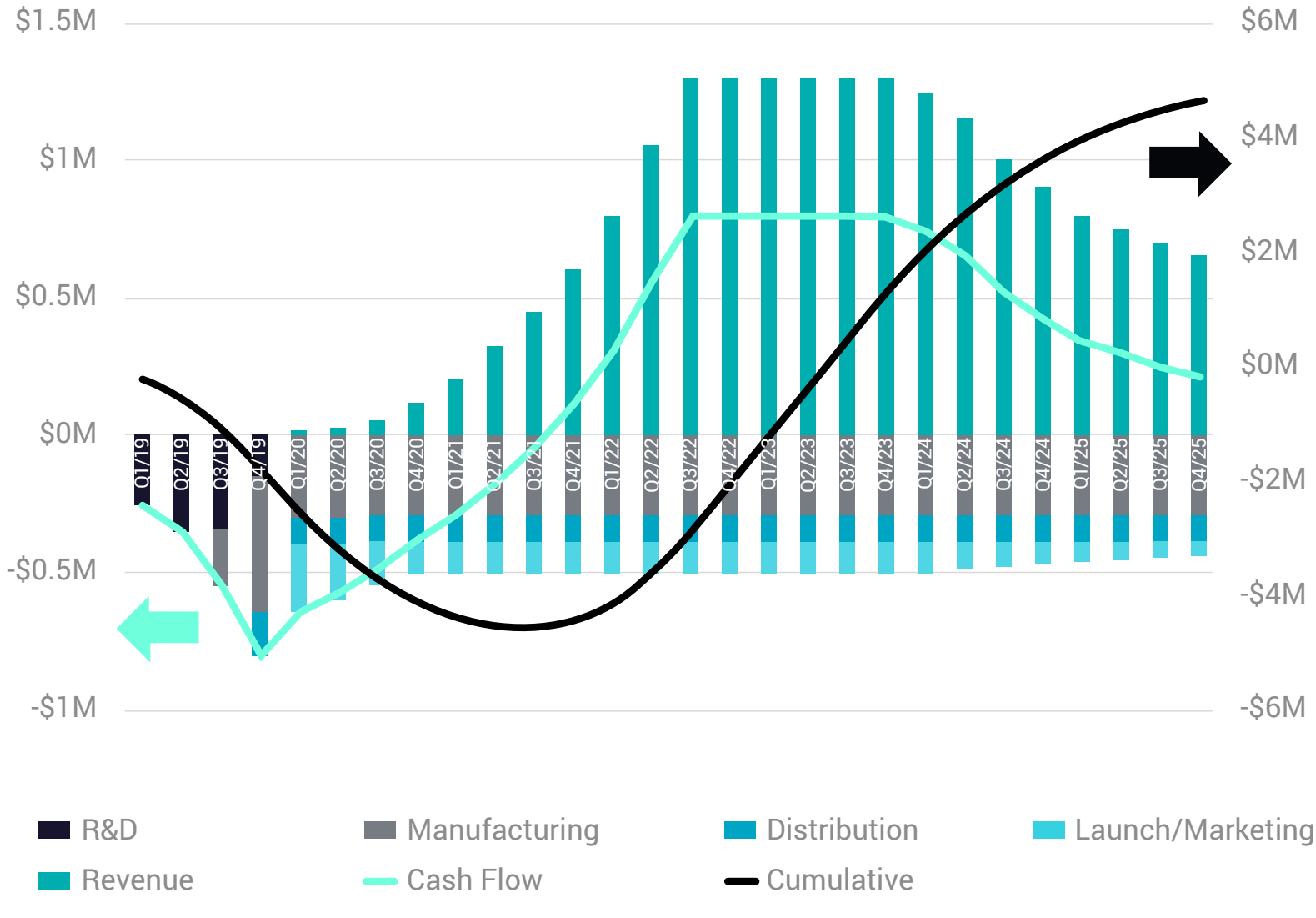
06

PROTOTYPE BUILD & PRODUCTION VOLUMES

Prototype build volumes throughout the development lifecycle (not just launch and steady-state production volumes) can significantly impact development cost. It's important to think ahead and develop prototype build plans based on needs, including yield considerations (yield is often low during prototype stages).

Material costs will be really high for products requiring hundreds or thousands of prototypes in early stages to enable things like data collection, reliability testing, regulatory testing, or developer community seeding. Beyond the costs for large numbers of units, each design-build-test iteration multiplies these costs. When these two factors combine with high BOM costs, we've seen material costs approach parity with design and development costs—imagine materials costing more than your engineering team before you even hit production!

Higher expected production volumes can motivate product cost optimization, since the tradeoff of saving even a few cents per unit can represent a significant savings in the long run at volume. Beyond product cost optimization, production volumes also have a big impact on manufacturing partner selection, including global location as well as tier (size and capability). Higher-volume products typically require larger investments in areas like tooling, test procedures, supplier qualifications, IQC/OQC, sourcing, and sustaining engineering.



Unfortunately, many hardware product companies reach a “valley of death” in the late stages of development. This happens when the cost of development has been higher than anticipated and there isn’t sufficient budget remaining to get through production to start generating revenue. Crowdfunding platforms in particular have seen many companies die in the valley. With this in mind, designing for ultra-high volume off the bat might not be the right approach for those without deep-enough pockets to match that ambition—remember to keep a minimum viable product (MVP) mentality. The “valley of death” is shown visually by the black cumulative cash balance line above.

< Contents

07

DEVELOPMENT BUDGET & SCHEDULE CONSTRAINTS

“ALL BUDGETS ARE FINITE, AS ARE THE BUSINESS CASES DRIVING THEM”

While we are seeking to answer the question of **“How much will it cost?”**, we can’t do this in a vacuum. All budgets are finite, as are the business cases driving them. It’s important to know our client’s constraints with respect to cost and time going into our assessment. If the client has especially tight constraints, we need to be able to present ways to tighten the scope.

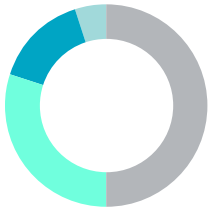
For example on scope, perhaps the form factor can change slightly, the launch volume can reduce, or a non-critical accessory can be removed. Schedules can be more difficult. In the consumer space, we’re often gunning for holiday launches. Sometimes rumors of competitive releases can also drive our schedules.

It can be easy to fixate only on the total development budget and launch date, but there are other important constraints to consider, such as interim funding milestones and the investment payback period. If we focus on making the end product, but don’t have a demo to show investors to enable fundraising before cash dries up, we might all be dead in the water.

Scoping and considering constraints is an iterative process and it's essential to go through it. Difficult decisions will always need to be made as we try our best to optimize scope, schedule, and budget. If we don't sufficiently consider these constraints, the cost and schedule will be scrutinized eventually (often too late to successfully course correct). If we over-constrain, the product might not support an interim funding milestone, make it to market, or be sufficiently compelling when it launches. Threading the needle takes time and iteration in up-front planning as well as ongoing project governance.

08

RISK TOLERANCE



Conservative

Balanced

Growth

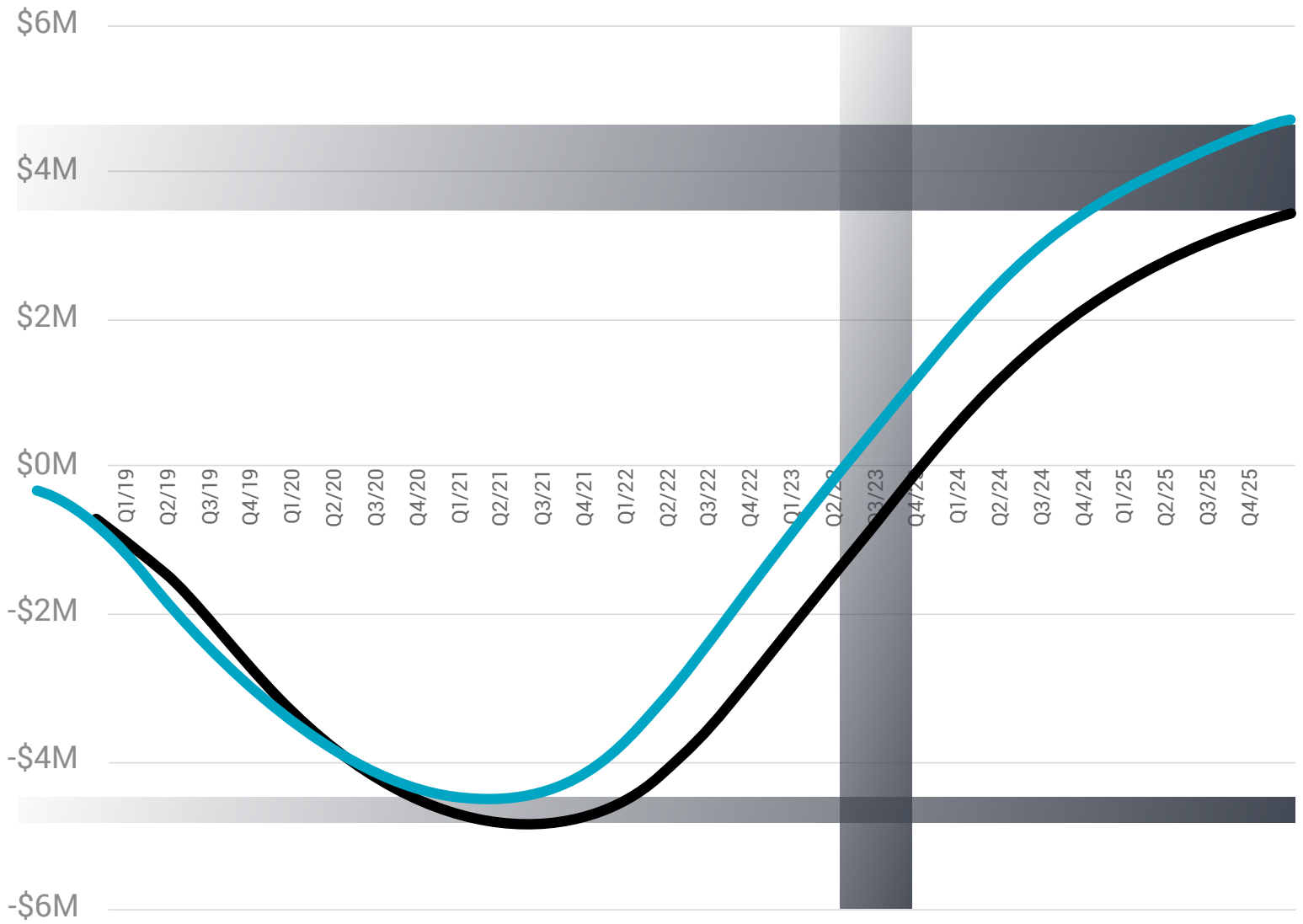
Aggressive Growth

Much like designing a financial investment portfolio, designing a product development strategy can be heavily impacted by risk tolerance. With risk comes increased potential for reward, but also the potential for downside.

Innovative product development is an inherently risky endeavor and requires some risk tolerance. That said, nobody wants to take on unnecessary risk.

The key for innovative product development is understanding and deliberately choosing to take on risk where it matters most (to differentiate the product, to get it to market faster, etc.) and attenuating risk elsewhere.

If a client is willing to accept more risk for the possibility of a lower-cost and faster program, we can look to cut certain corners and design a “happier path” development strategy. The reward comes when things go well and this aggressive approach pays off with fewer iterations, a shorter schedule, and a lower overall cost than might have been the case with a more measured approach. The risk comes when Murphy’s Law makes an appearance and the happy path is no longer valid, or when an overly-aggressive strategy hamstrings solid engineering and results in needing to scrap big chunks of development effort. Big risks coming to fruition often leave insufficient time, budget, or both to complete the project.



We like the concept of a separated contingency in product development plans. We can design a baseline plan which balances optimism and pessimism to match the client's risk tolerance, but also agree with our client on a contingency level they should hold. If we don't need it, we won't charge for it. But it's naive to expect everything to go perfectly. The simple mechanism of having a separate contingency can facilitate efficient development without resulting in a funding gap to see the project through when things don't go right.

The visual above shows how much a 3-month delay can impact a hypothetical program's funding needs (up $350k), break-even point (4 months later), and total payoff ($1M less). Planning for the best and being prepared for some bumps in the road isn't just prudent, it might be the difference between success and failure.

PUTTING IT ALL TOGETHER

As the crowdfunding explosion and subsequent retraction has demonstrated, hardware development is hard. There are real risks and iteration is inherently more time-consuming and expensive than pure software development. At Synapse, with our collective experience of developing hardware products for clients, large and small, over nearly two decades, we've built up a lot of expertise in estimating programs and can steer clients around pitfalls.

With all of this knowledge, Synapse can easily fall into a trap. If we burden our estimates for new clients too heavily with the collective baggage of our past experiences (called reference-class forecasting), we can come across as inefficient and overly expensive. This is especially true when working with clients who haven't yet been through hardware development cycles and can bring an optimistic bias. We don't want to spin a story to our clients and tell them up front that it'll be easier and faster than we think it'll really be in the end—this could result in them never making it through the valley of death. So, we have to balance these competing forces to prepare our clients for the journey while still designing and holding ourselves to the most efficient path possible.

The best way we've found to work through this with our clients is to assess and dig into the levers I've outlined, iterating on them together to find a strategy that works both up front and throughout the project, often using the concept of a client-owned contingency budget for handling risks. Each client and each project has its own priorities, constraints, and risk tolerance.

Successful projects are the most fun and impactful. **Success comes from our ability to partner with our clients, tailoring our approach together and pulling these levers accordingly.** We hope you're able to use this guide to achieve your product development goals while having fun along the way!

< Contents

SYNAPSE

Seattle
1511 6th Ave
Seattle, WA 98101
Ph: 206.381.0898

San Francisco
640 Bryant St
San Francisco, CA 94107
Ph: 415.361.5088

www.synapse.com

Questions? Project Ideas?
Give Dylan a shout.

Dylan Garrett
VP Consumer & Industrial Business

dylan.garrett@synapse.com