PARTE I — ENTENDER EL PRODUCTO · CAPÍTULO 01

¿Qué le puede enseñar el diseño industrial a los líderes de producto digital?

Materiales, tolerancias, costo y mantenimiento definen todo objeto físico mucho antes de llegar a un usuario. Este capítulo argumenta que un producto es 10% idea y 90% esfuerzo por demostrar que esa idea puede sobrevivir a la realidad — en madera o en software.

Publicado el 9 sep 2026 · 18 min de lectura · Edición 0.1 · Estado: Publicado

Hay una parte del diseño de producto que ocurre mucho antes de que exista algo que podamos tocar, descargar o poner frente a un usuario — menos vistosa que el render o la primera pantalla funcional, pero probablemente donde se decide gran parte de lo que el producto llegará a ser: costo, fabricación, mantenimiento, vigencia.

Cuando estudiaba diseño industrial aprendí a convivir con esas preguntas antes de acostumbrarme a llamarlas estrategia de producto. Años después, trabajando con productos digitales, plataformas y portafolios empresariales, las mismas preguntas volvieron a aparecer con otros nombres. Cambiaron los materiales; no cambió demasiado el problema.

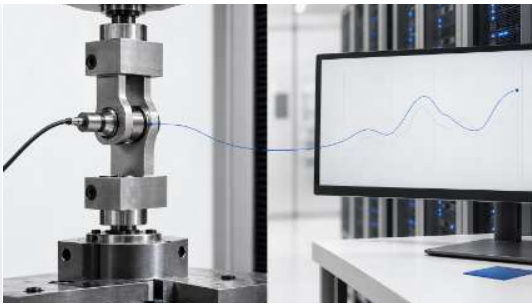
Si tuviera que reducirlo a una proporción deliberadamente imperfecta: un producto es 10% idea y 90% esfuerzo por demostrar que esa idea puede sobrevivir a la realidad. No es una estadística — es una forma de ordenar la experiencia que este capítulo despliega a través de una silla de 1859, algunas hipótesis incómodas y el argumento de que la arquitectura digital es, ella misma, un material de diseño.



Seis piezas, diez tornillos, dos tuercas: la arquitectura que Thonet resolvió en 1859.

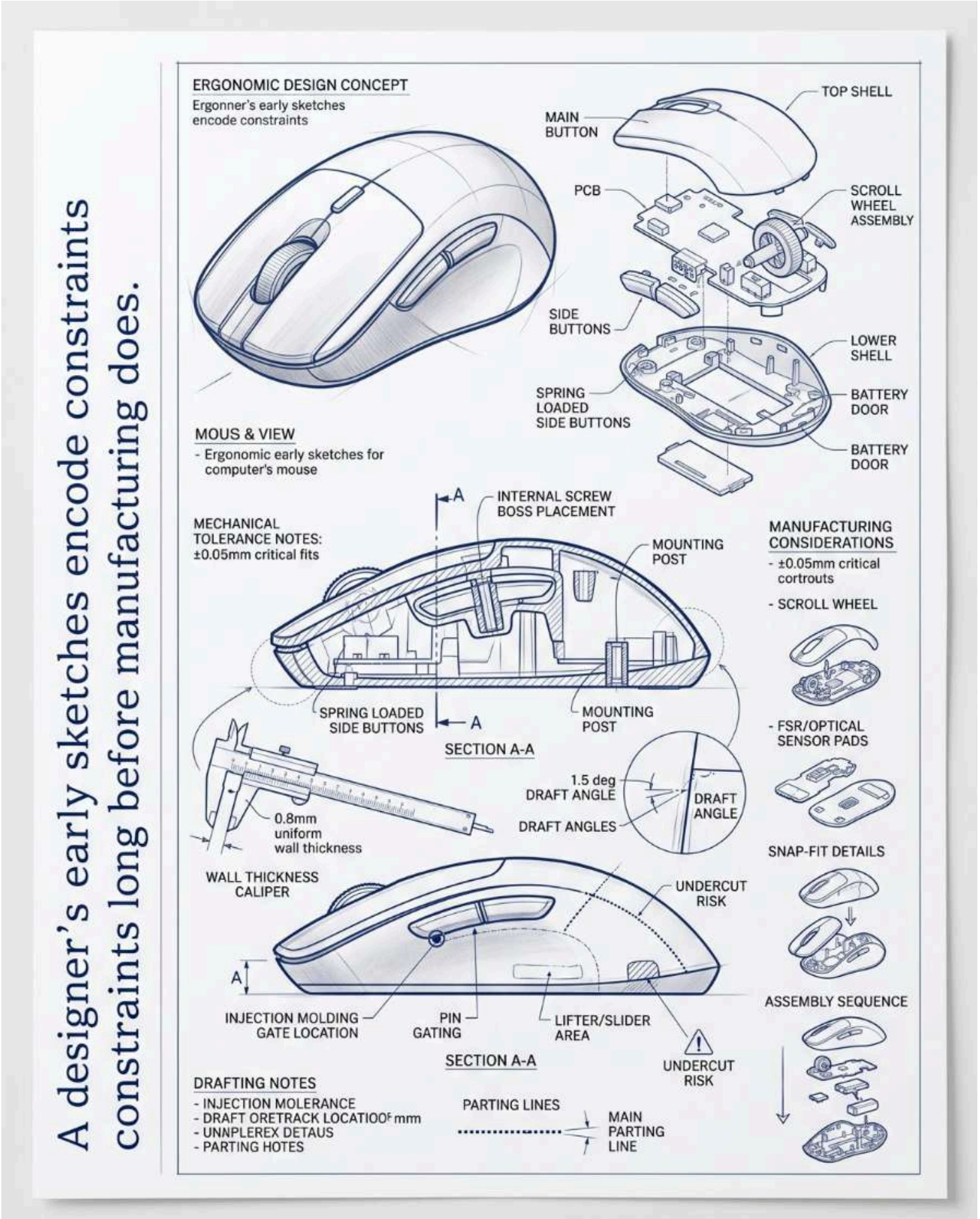


El costo de las herramientas es una decisión que se toma una vez y se paga en cada unidad después.



Una bisagra bajo carga se comporta como una API bajo tráfico: ambas tienen un punto de fluencia.

La idea es apenas una hipótesis



Los primeros bocetos de un diseñador codifican restricciones mucho antes que la manufactura: tolerancias, ángulos de desmolde, espesores de pared.

Porque una idea no tiene todavía costo unitario, ni arquitectura, ni proveedores, ni canales, ni riesgo, ni retorno esperado. Tampoco tiene soporte, capacidad de producción, backlog, métricas de adopción o un modelo claro para capturar valor. Todo eso aparece cuando dejamos de enamorarnos de la posibilidad y comenzamos a diseñar el producto. En ese sentido, el desarrollo físico y el digital parten de una condición parecida: antes de construir hay que convertir incertidumbre en decisiones.

Existe una tentación frecuente en producto: confundir la claridad de una idea con su viabilidad. En diseño industrial el error se descubre rápido porque la materia opone resistencia. Una forma puede ser atractiva y al mismo tiempo imposible de fabricar con el proceso elegido; puede requerir un molde demasiado costoso para el volumen esperado; puede pesar demasiado para el canal logístico; puede usar una unión elegante que vuelve prohibitiva la reparación.

El objeto obliga a negociar.

El software concede un poco más de tiempo antes de cobrar la cuenta. Podemos construir una experiencia que funcione durante una demostración y todavía no tener un producto capaz de operar: faltan procesos de identidad, permisos, manejo de excepciones, telemetría, facturación o soporte. La ausencia de material físico no elimina las restricciones; simplemente las vuelve menos visibles.

Ulrich, Eppinger y Yang estructuran el desarrollo como una disciplina interdisciplinaria en la que conviven oportunidad, planeación, necesidades del cliente, especificaciones, conceptos, arquitectura, diseño industrial, manufactura y cadena de suministro, prototipado, economía del producto y administración del proyecto. Robert Cooper, desde Stage-Gate, coloca antes del desarrollo a escala una investigación inicial de mercado y factibilidad y, después, un business case donde se cruzan atractivo comercial, factibilidad técnica, riesgo y retorno financiero.

Me interesa menos defender un método específico que señalar algo que ambos modelos dejan claro: el desarrollo de producto empieza bastante antes del desarrollo técnico. ¿Existe realmente una necesidad o solamente tenemos una buena historia para contarla? ¿El problema es suficientemente frecuente o costoso para que alguien cambie de comportamiento? ¿Cuánto tendríamos que invertir antes de aprender si estábamos equivocados?

En diseño industrial esas preguntas desembocan en materiales, proveedores, herramental, volúmenes y precio. En un producto digital se convierten en estimaciones de desarrollo, nube, licenciamiento, datos, adquisición de usuarios y evolución tecnológica. Las hojas de cálculo cambian. La decisión económica sigue siendo esencialmente la misma.

“La monetización modifica el producto.”

Un producto no existe porque pueda construirse. Existe cuando encontramos una forma razonable de crear valor, entregarlo y capturar suficiente valor para sostener el sistema que lo produce. David Teece define el modelo de negocio como la arquitectura mediante la cual una empresa crea, entrega y captura valor — una definición que devuelve la monetización al lugar donde debería estar: dentro del diseño del producto, no como una conversación tardía.

No es lo mismo diseñar algo que se vende una sola vez que algo que deberá justificar una suscripción cada mes. No es igual vender una máquina que vender el servicio que esa máquina produce. Tampoco es igual ofrecer software por licencia, por consumo, por asiento, por transacción o mediante un esquema combinado. Cada decisión termina afectando arquitectura, experiencia, métricas, operación y relación con el cliente.

En un objeto físico es más fácil verlo porque el costo de cada unidad está frente a nosotros. En digital existe la tentación de pensar que, como copiar software cuesta muy poco, la economía es trivial. Pero el costo marginal bajo no significa costo total bajo: disponibilidad, almacenamiento, procesamiento de transacciones, modelos de inteligencia artificial, infraestructura, incidencias y adquisición de usuarios también forman parte del producto.

Ahí comienza a romperse otra falsa frontera: la que separa demasiado pronto diseño, tecnología y negocio.

El costo que se congela antes de vender la primera unidad



Diseñar es decidir bajo restricciones



El material es una restricción entre muchas — costo, escala, tiempo, regulación, canal.

En la escuela uno puede enamorarse de la forma. Es natural. Pero llega un punto en que el producto deja de ser únicamente aquello que queremos que sea y empieza a convertirse en aquello que puede ser dentro de un conjunto de restricciones. El material es una de ellas, no la única: también lo son el costo, el proceso, la escala, el tiempo, la regulación, la capacidad del proveedor, el empaque, el canal y la expectativa del usuario. Esas restricciones no llegan al final para corregir el diseño. Participan de su definición.

En producto digital sucede lo mismo. Arquitectura, datos, identidad, seguridad, interoperabilidad, latencia, disponibilidad, privacidad o capacidad de integración no deberían ser una lista de problemas que tecnología resuelve después de que producto terminó de imaginar la experiencia. Son materiales de diseño.

Cuando digo que la arquitectura digital es un material no lo digo como metáfora estética. Tiene propiedades y límites. Una API puede soportar cierto patrón de uso y no otro. Un modelo de datos puede hacer sencilla una clase de evolución y convertir otra en una migración costosa. Una decisión de identidad puede permitir compartir capacidades entre productos o encerrarlos en dominios separados.

El diseñador industrial aprende a preguntar qué permite hacer el material. El product manager debería hacer una pregunta parecida sobre el sistema sobre el que está construyendo. Y esto importa porque el producto que llega al mercado no es solamente la suma de funcionalidades visibles. Es el resultado acumulado de decisiones previas. Algunas abren posibilidades. Otras las cierran.

6	36	50M+
PIEZAS PRINCIPALES — MÁS 10 TORNILLOS Y 2 TUERCAS	SILLAS DESARMADAS POR METRO CÚBICO EMPACADO	UNIDADES PRODUCIDAS PARA 1930

Una silla que en realidad era un sistema



La silla No. 14 de Thonet, comercializada hoy como modelo 214.

Hay una silla que resume gran parte de este argumento mejor que cualquier diagrama contemporáneo. Michael Thonet desarrolló durante la primera mitad del siglo XIX técnicas para curvar madera y terminó perfeccionando durante la década de 1850 un proceso de doblado de madera maciza que hizo posible producir mobiliario en serie. En 1859 apareció la silla que entonces se conoció como No. 14 y que hoy Thonet comercializa como modelo 214.

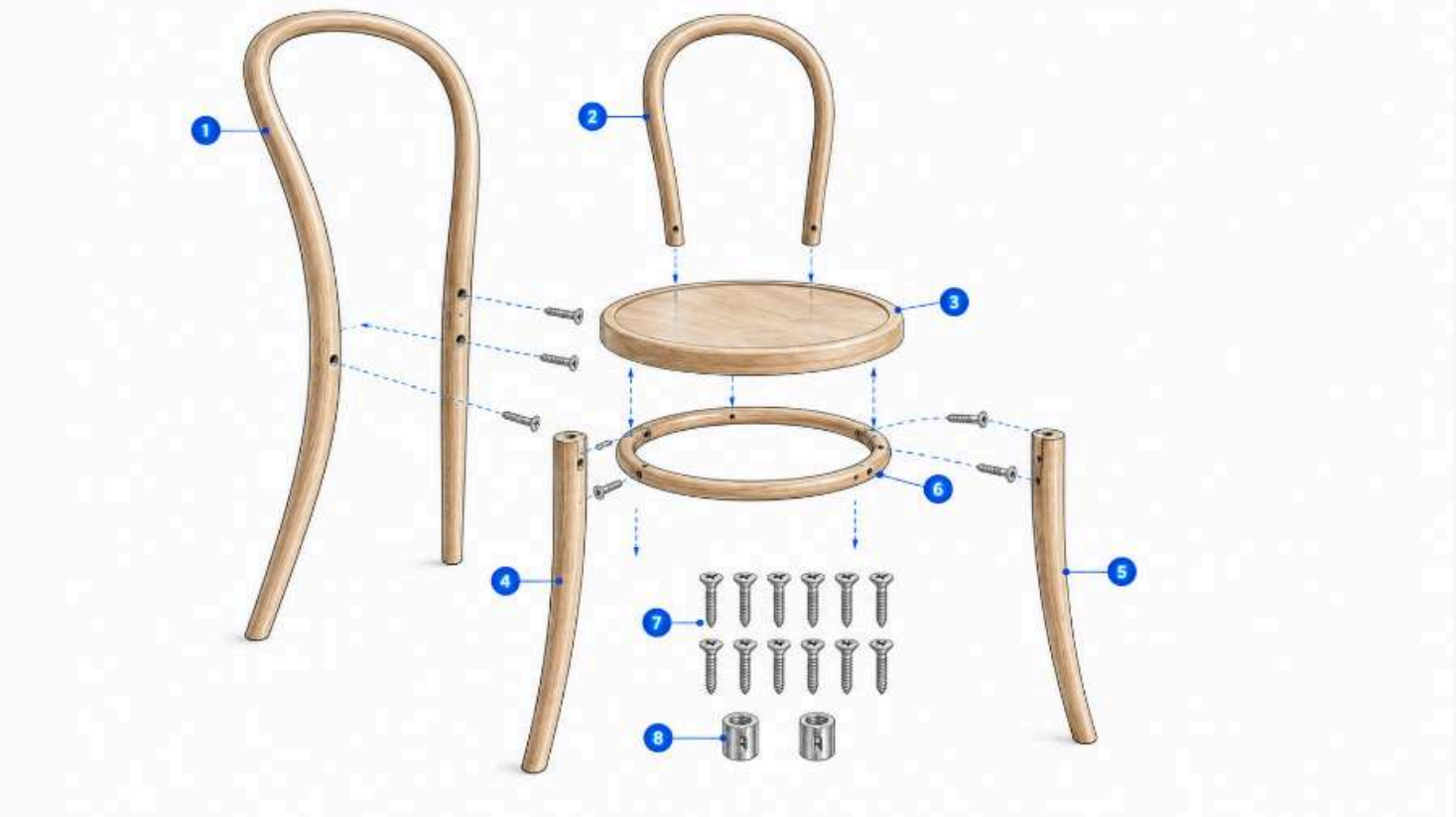
A primera vista su éxito podría atribuirse a la forma: ligera, reconocible, suficientemente neutral para existir durante décadas en cafés, restaurantes y hogares. Pero concentrarse únicamente en la silueta pierde la parte más interesante. La silla era un sistema de decisiones.

El modelo podía resolverse con seis piezas principales, diez tornillos y dos tuercas. Esa reducción no era únicamente formal. Permitía separar fabricación y ensamblaje, estandarizar componentes, reducir trabajo artesanal y transportar la silla desmontada. Thonet documenta que 36 sillas desarmadas podían empacarse en una caja de un metro cúbico y ensamblarse en destino. Para 1930 se habían producido más de 50 millones de unidades.

En otras palabras: la silla no fue diseñada solamente para sentarse. Fue diseñada para producirse, repetirse, empacarse, viajar y ensamblarse. Eso cambia completamente la lectura del objeto.

Hay una decisión tecnológica —el doblado de madera—, una decisión de componentes —pocas piezas estandarizadas—, una decisión de unión —tornillos y tuercas—, una decisión de manufactura —producción repetible— y una decisión logística —transportar volumen

Seis piezas, un sistema



Vista explosionada: seis piezas de madera curvada más el herraje que las une.

Ilustración en diseño

Caja de embalaje de un metro cúbico: 36 sillas desmontadas listas para ensamblar en destino.



Detalle del doblado de madera maciza que Thonet perfeccionó en la década de 1850.



Tornillo y tuerca: la unión que hace reparable — y repetible — al sistema completo.

Arquitectura que se puede repetir



Una plataforma es una colección de activos compartidos por un conjunto de productos.

Esa es una de las razones por las que me parece un antecedente tan útil para entender el producto digital. No porque una API sea equivalente a una pata de madera, sino porque el principio de diseño es reconocible: si una arquitectura permite reutilizar componentes,

entonces la siguiente variante cuesta menos que inventar nuevamente el sistema completo.

La investigación contemporánea sobre arquitectura modular describe justamente ese problema. Dahmus, Gonzalez-Zugasti y Otto analizan familias de producto construidas con módulos compartidos e intercambiables, buscando reutilizar funciones entre variantes. Robertson y Ulrich describen una plataforma como una colección de activos —componentes, procesos, conocimiento, personas y relaciones— compartidos por un conjunto de productos.

Si uno elimina por un momento el vocabulario industrial, es difícil no reconocer una conversación moderna de plataformas digitales. Una capacidad de identidad común puede alimentar varias experiencias. Un servicio de pagos puede utilizarse en diferentes productos. Un catálogo, una capa de datos o una librería de componentes pueden convertirse en activos compartidos. Lo importante no es reutilizar por reutilizar; es que la arquitectura permita producir variedad sin multiplicar proporcionalmente el costo y la complejidad.

> *"Thonet entendió esto en madera. Muchas compañías todavía están intentando entenderlo en software."*

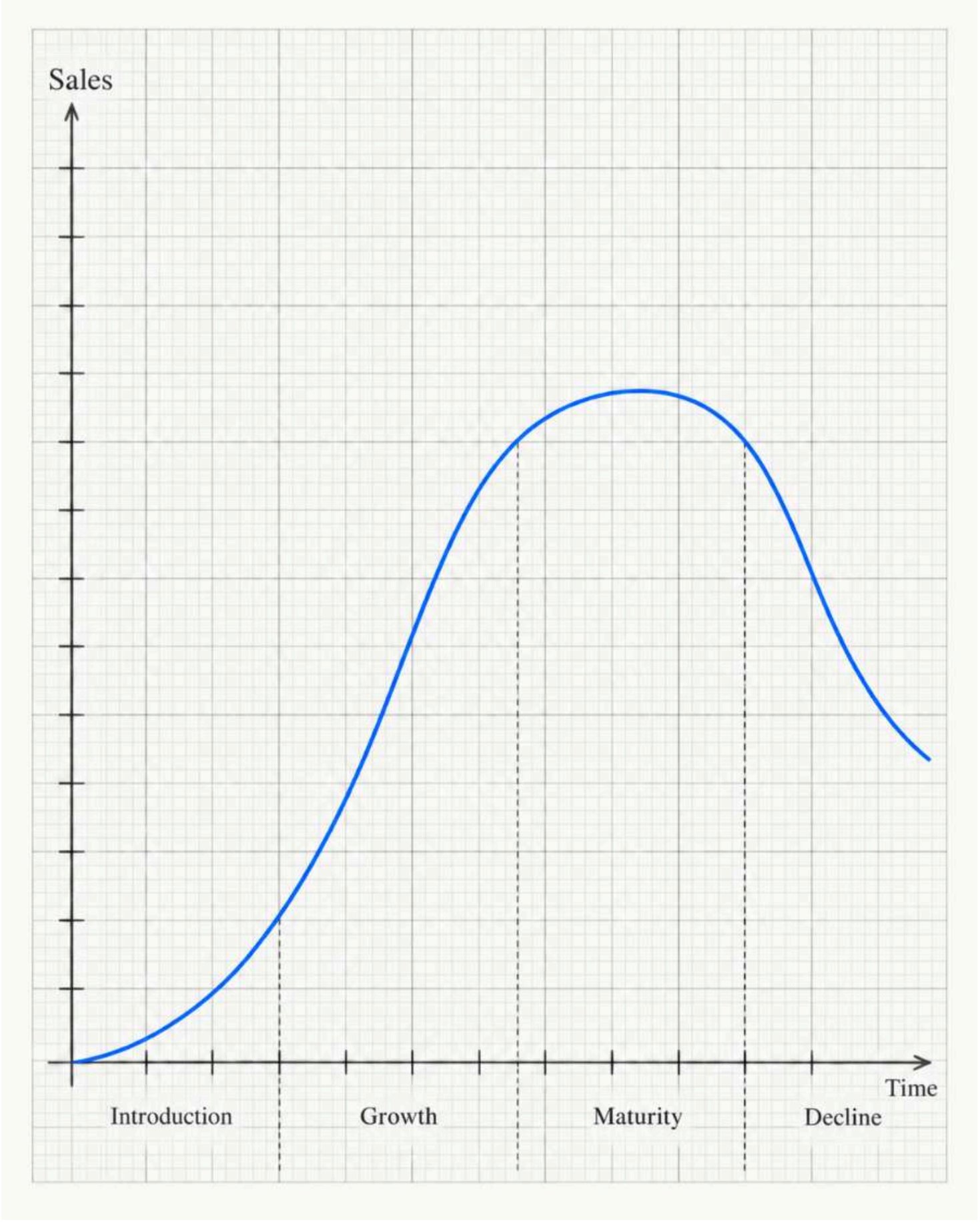
Otro aprendizaje del producto físico es que una versión no necesita contener todo lo que el producto podría llegar a ser. La industria ha trabajado durante mucho tiempo con generaciones, variantes, plataformas y mejoras incrementales. A veces olvidamos esto cuando contamos la historia del producto digital como si el concepto de iteración hubiera nacido con Agile. No fue así.

Un estudio de la U.S. Government Accountability Office sobre prácticas de desarrollo encontró que las compañías comerciales analizadas reducían riesgo utilizando desarrollo evolutivo: limitaban la cantidad de contenido nuevo en una generación y trasladaban capacidades aún inmaduras a generaciones posteriores, porque los cambios se vuelven más costosos conforme el programa avanza.

La iteración no debería ser excusa para comenzar sin criterio; debería ser una estrategia para decidir qué necesitamos aprender ahora y qué puede esperar hasta que tengamos evidencia. El objetivo de una primera versión no es fabricar una versión pobre de la visión completa. Es construir una versión suficientemente coherente para probar las hipótesis que más podrían destruir el caso si resultaran falsas.

¿La gente entiende el valor? ¿Puede completar la tarea? ¿Regresa? ¿Paga? ¿La arquitectura soporta el patrón real de uso? Cada producto debería tener sus propias preguntas. Ahí es donde incremental no significa pequeño. Significa controlado.

El ciclo de vida no empieza en el lanzamiento



El lanzamiento no inaugura el producto. Lo expone.

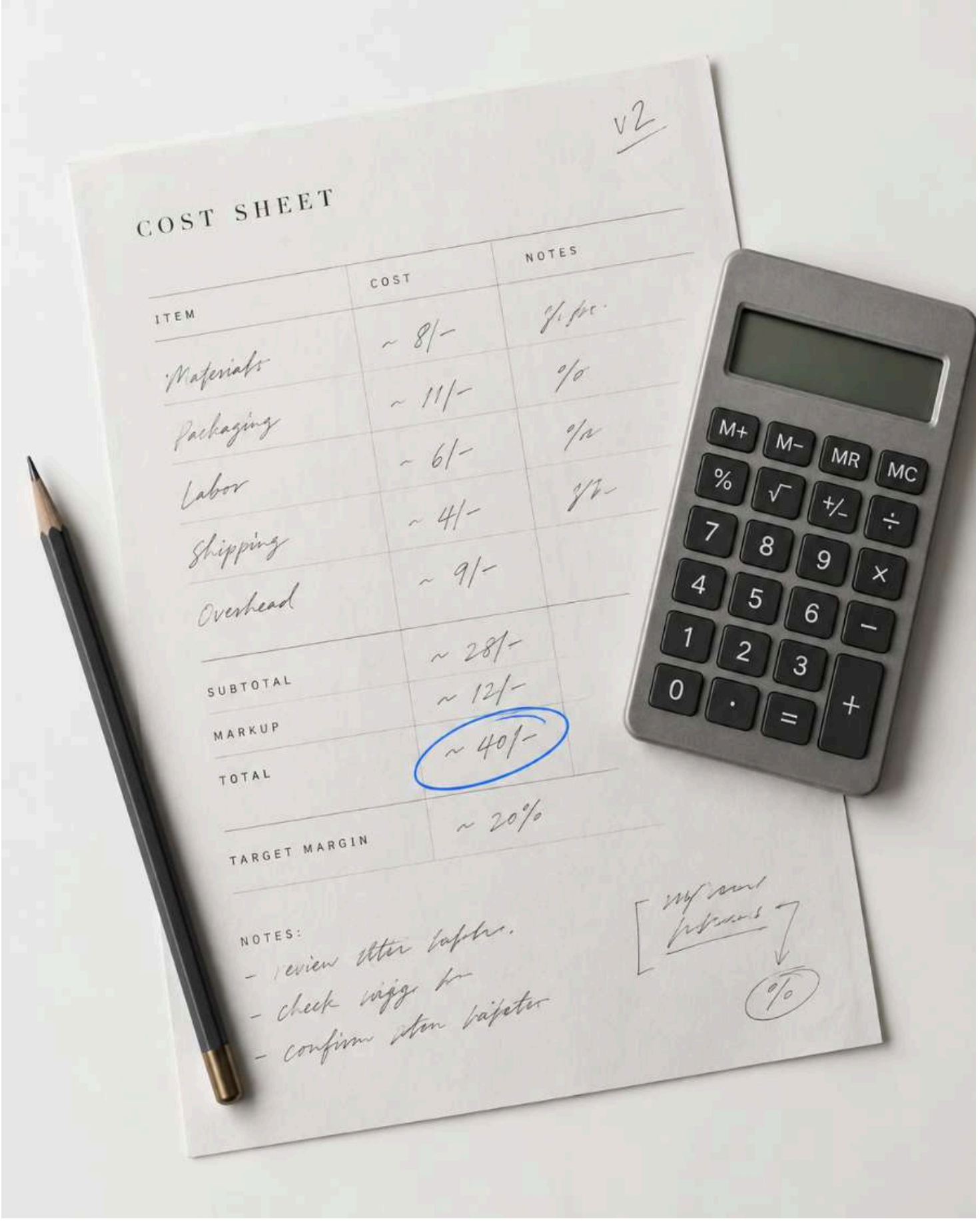
Durante mucho tiempo, el gráfico clásico del ciclo de vida del producto se representó como una curva que atraviesa introducción, crecimiento, madurez y declive. Theodore Levitt convirtió esa idea en herramienta de estrategia en 1965. La curva es útil, pero puede crear una interpretación demasiado cómoda si pensamos que el trabajo de producto comienza en la izquierda del gráfico, justo en el lanzamiento.

Para cuando un producto llega ahí ya se tomaron muchas de las decisiones que limitarán su trayectoria: estructura de costos, cadena de suministro, arquitectura de componentes en lo físico; infraestructura, modelo de datos, adquisición, pricing en lo digital.

El lanzamiento no inaugura el producto. Lo expone. Es el momento en que las hipótesis acumuladas se encuentran simultáneamente con usuarios, operación y economía real. Stage-Gate describe el lanzamiento como el paso hacia operaciones o producción a escala; en el modelo de Ulrich y Eppinger, el desarrollo físico tampoco termina en el diseño detallado, sino que incluye pruebas, refinamiento y production ramp-up.

El equivalente digital es evidente. Un producto que funciona con cien usuarios puede no funcionar con cien mil. Una operación que soporta manualmente diez excepciones puede colapsar con diez mil. Antes del lanzamiento intentamos responder si debemos y pod

La rentabilidad que diseño no debería delegar



El diseño es una negociación entre variables — también las económicas.

Uno de los puntos de encuentro que más me interesan entre diseño industrial y producto digital es la economía. Mi experiencia es que la economía es parte del material de diseño: si un producto requiere una inversión alta de tooling, necesitamos un volumen capaz de amortizarla; si una pieza compleja puede convertirse en tres piezas sencillas, quizá aumentemos el ensamblaje pero reduzcamos desperdicio o mantenimiento.

En digital pasa igual. Podemos ofrecer una capa gratuita para adquirir usuarios y cobrar por capacidades avanzadas; podemos cobrar por asiento y descubrir después que el valor depende de volumen transaccionado; podemos combinar suscripción y consumo. No existe un modelo universal. Lo que sí existe es una relación inevitable entre cómo capturamos valor y qué producto terminamos construyendo.

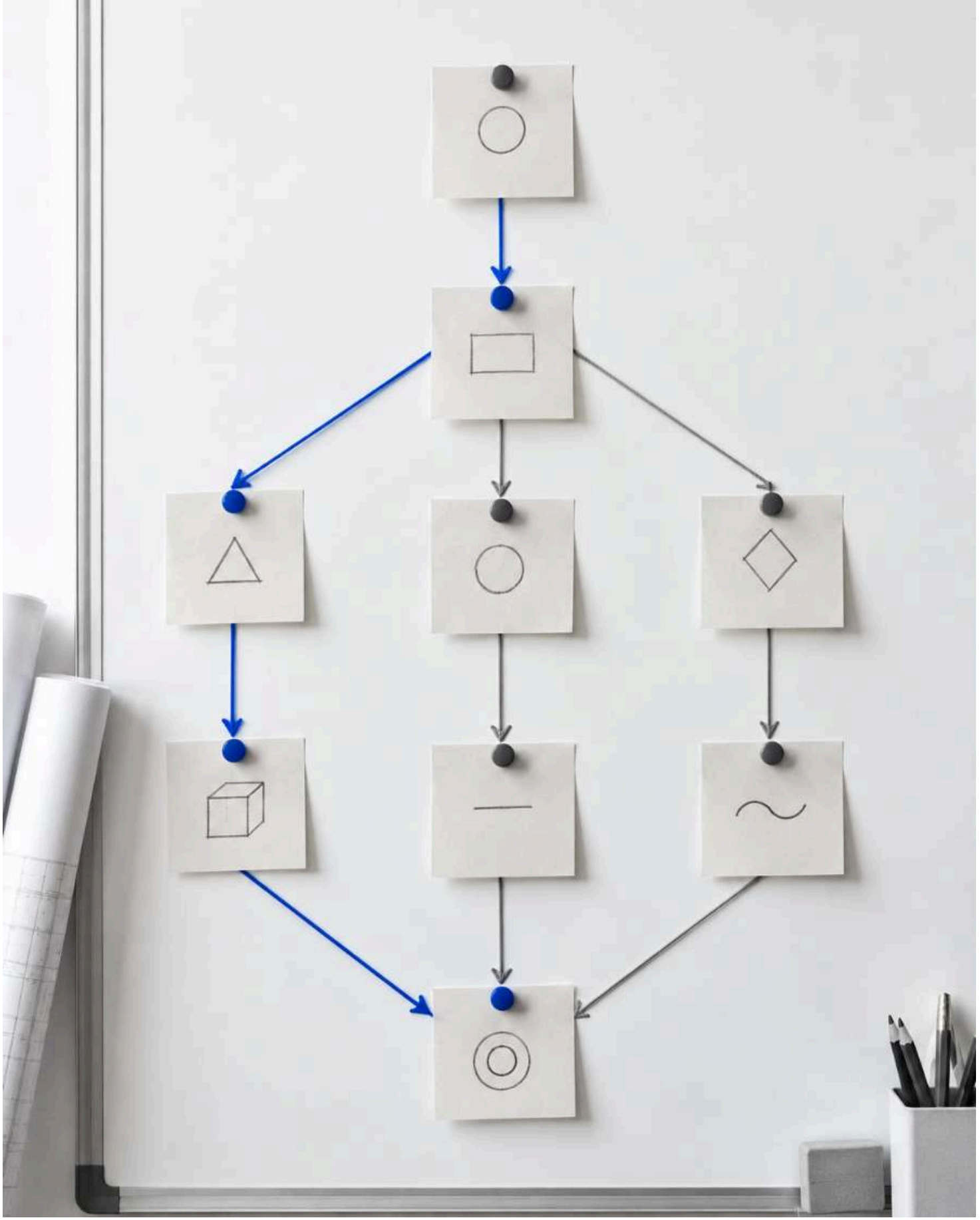
Por eso un plan de monetización no debería ser una lámina para inversionistas. Debería convertirse en hipótesis explícitas: quién paga, por qué paga, cuándo paga, por qué seguiría pagando y qué costos crecen cuando crece el uso.

En el producto físico la venta hace visible de inmediato el intercambio económico. En digital podemos acumular uso durante meses antes de descubrir si el comportamiento se traduce en ingresos suficientes. Un usuario no es todavía un modelo de negocio. Una descarga tampoco. Y una idea viral puede seguir siendo una mala empresa.

Fluencia física, saturación digital



Producto y proyecto: familiares, no idénticos



Gestionar producto es saber dónde necesitamos control y dónde necesitamos aprendizaje.

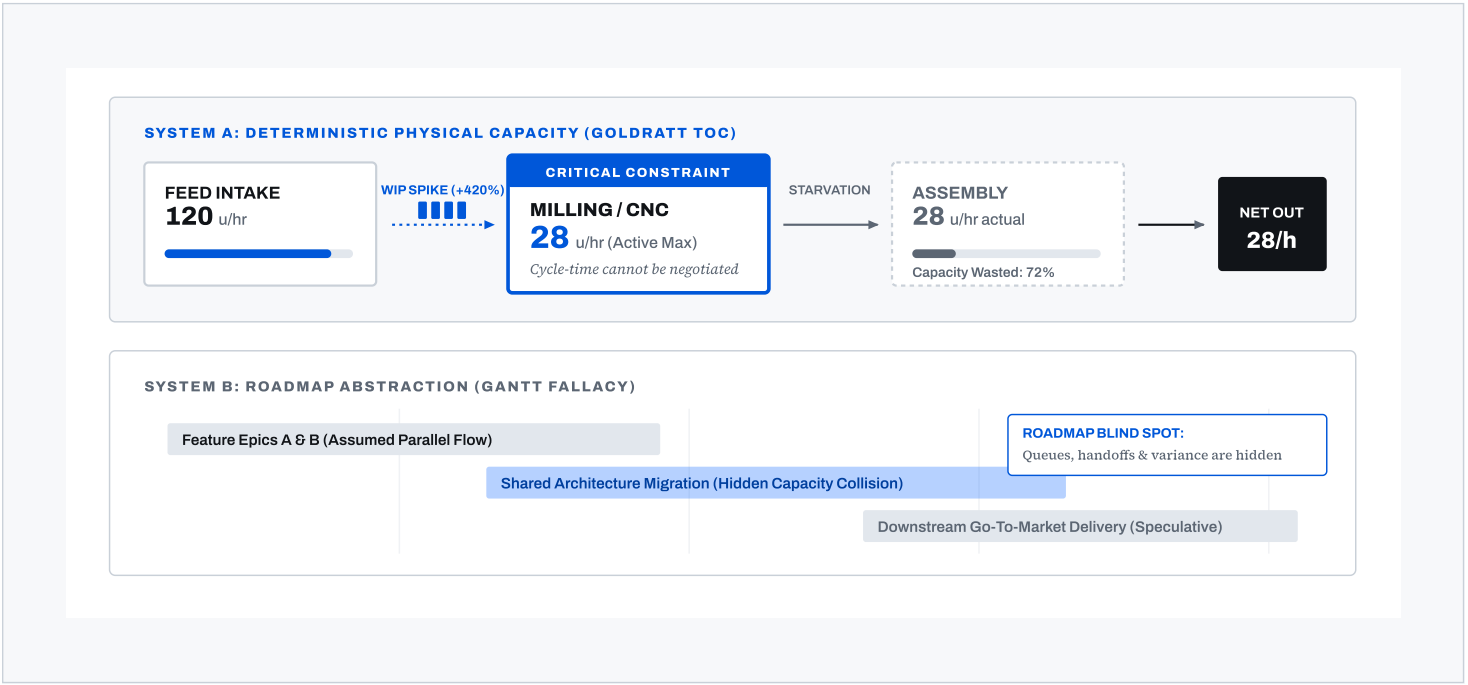
Un proyecto existe para producir un resultado dentro de una combinación particular de alcance, tiempo, recursos, dependencias y riesgo. El producto, en cambio, puede atravesar decenas de proyectos durante su vida: lanzarlo puede requerir uno, migrar su arquitectura otro, entrar a un mercado distinto un tercero.

La investigación de Jetter, Albar y Sperry para PMI estudió cómo se complementan las prácticas de project management y desarrollo de producto en entornos tecnológicos. Su conclusión no fue que el control tradicional deba imponerse a la innovación, sino que los mecanismos de gestión necesitan adaptarse al nivel de incertidumbre — una distinción más útil que la vieja discusión entre "Agile" y "Waterfall".

No todos los elementos de un producto tienen el mismo nivel de incertidumbre. Puedo tener una fecha contractual completamente determinista y una solución técnica todavía abierta; una arquitectura madura y una nueva regulación que cambia la economía. Gestionar producto consiste también en saber dónde necesitamos control y dónde necesitamos aprendizaje.

En diseño industrial esto se vuelve evidente hacia producción: hay elementos que deben quedar congelados antes de invertir en herramental o escala. En software tenemos mayor libertad para cambiar después del lanzamiento, pero esa libertad no es infinita — migrar datos, romper compatibilidad o reemplazar componentes centrales también puede ser extraordinariamente caro.

La restricción visible, el roadmap que la esconde



Lo que una silla de 1859 aún enseña



Su valor no estaba solamente en el objeto terminado sino en el sistema capaz de repetirlo.

La silla de Thonet condensa casi todo el capítulo. No comenzó siendo únicamente una forma atractiva: necesitó una tecnología de fabricación madura, una arquitectura de pocas piezas, un método de unión, repetibilidad, una estrategia logística. Y después necesitó la capacidad de usar principios y componentes comunes para producir variaciones sin comenzar de cero.

En producto digital nos obsesionamos con frecuencia con el lanzamiento visible: la aplicación, la pantalla, la nueva funcionalidad. Pero los productos capaces de sobrevivir suelen depender de algo menos visible: una arquitectura que pueda evolucionar, una economía que pueda sostenerse, una operación que pueda repetirse y una organización que pueda aprender más rápido de lo que acumula complejidad.

Ambos parten de necesidades imperfectamente conocidas. Ambos convierten ideas en especificaciones. Ambos deciden una arquitectura. Ambos enfrentan restricciones técnicas y económicas. Ambos necesitan prototipos, resolver producción u operación, y llegan a un mercado que puede ignorarlos. Ambos terminan desapareciendo si el valor que producen deja de justificar el costo de mantenerlos.

La diferencia más grande probablemente no está en la naturaleza del producto sino en la velocidad y el costo con que podemos corregir ciertas decisiones. El software nos permite modificar muchas cosas después de lanzar — esa ventaja también puede volvernos descuidados, hasta que descubrimos que también existe tooling digital: contratos, datos, integraciones, procesos y dependencias difíciles de cambiar. El producto físico nos obliga a respetar esas consecuencias antes. Tal vez por eso todavía tenga mucho que enseñarnos.

Cuatro pasos para nombrar una tolerancia

Un método breve, usado a lo largo de este libro, aplicado aquí por primera vez a una sola decisión de producto.

1

Nombra la restricción

Enuncia el límite real — costo, latencia, personal — en una frase.

2

Escribe el número

Asígnale un valor medible, aunque sea aproximado, no un adjetivo.

3

Asigna un responsable

Una persona o equipo que responda cuando se exceda.

4

Verifica antes de lanzar

Compara contra el número, no contra cómo se sintió la demo.

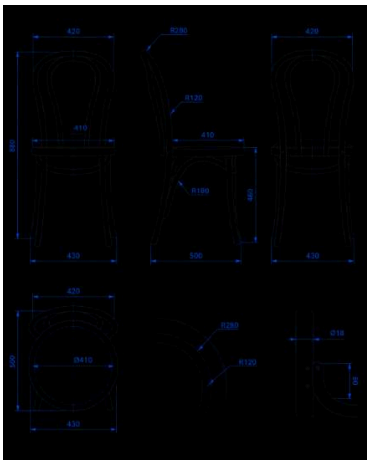
> *“Hacer que una idea pueda existir más de una vez.”*
>
> — J. A. Sosa

Cuando pienso en los productos que he visto construirse durante los últimos años —físicos, digitales, plataformas, capacidades empresariales— cada vez me interesa menos la idea inicial y más la cadena de decisiones que consiguió convertirla en algo sostenible. La idea importa. Sin ella no comienza nada. Pero rara vez es la parte escasa.

Lo escaso es encontrar una necesidad suficientemente importante, traducirla en una propuesta clara, demostrar que podemos construirla, diseñar una arquitectura que no nos condene demasiado pronto, encontrar una economía capaz de sostenerla, coordinar a las personas necesarias para llevarla al mercado y conservar suficiente flexibilidad para aceptar que alguna de nuestras primeras suposiciones será incorrecta. Ahí aparece mi 10/90: no como fórmula académica sino como recordatorio de que el entusiasmo inicial representa una fracción muy pequeña del camino.

El diseñador industrial hace esa negociación con materiales, procesos, proveedores, costos y manufactura. El líder de producto digital la hace con tecnología, datos, comportamiento, operación y economía. Los dos están intentando resolver el mismo problema fundamental.

Un prototipo demuestra que algo puede existir una vez. Un producto demuestra que podemos repetir el valor. Y un sistema bien diseñado permite que ese producto cambie sin tener que volver a inventarlo completo. Ahí comienza el siguiente capítulo.



Espacio de objeto transparente: contain, nunca cover — el diagrama completo permanece visible sin importar el ancho del contenedor.

REFERENCIAS

1. Ulrich, K., Eppinger, S., Yang, M. — *Product Design and Development.*
2. Cooper, R. — Stage-Gate: prácticas de desarrollo de nuevos productos.
3. Teece, D. — Definición del modelo de negocio como arquitectura de creación, entrega y captura de valor.
4. Thonet — Documentación histórica de la silla No. 14 / modelo 214, técnicas de doblado de madera y embalaje.
5. Dahmus, J., Gonzalez-Zugasti, J., Otto, K. — Investigación sobre arquitectura de plataformas y familias de producto modulares.
6. Robertson, D., Ulrich, K. — Planning for Product Platforms.
7. U.S. Government Accountability Office — Prácticas comerciales de desarrollo evolutivo y reducción de riesgo.
8. Levitt, T. (1965) — Exploit the Product Life Cycle.

9. Jetter, A., Albar, F., Sperry, R. — Investigación PMI sobre project management y desarrollo de producto en entornos tecnológicos.
10. Notas de campo del autor en gestión de programas, POS de retail y portafolios empresariales, 2016–2026.

Portfolio Health Diagnostic

¿Qué tan robusta es la arquitectura operativa de tu portafolio de producto? Responde 3 preguntas rápidas para descubrir tu nivel de madurez (Spore Seed diagnostic).

1. Asignación de Capacidad (Capacity Allocation)

¿Cómo decide tu equipo en qué proyectos trabajar el próximo trimestre?

- ○ Por urgencia (quien grite más fuerte o el cliente más enojado).
- ○ Por ROI estimado en hojas de cálculo que rara vez se cumplen.
- ○ Por cubetas estratégicas (Strategic Buckets) alineadas a los OKRs de la empresa.

Siguiente