

# **TECHNICAL WEIGHT**

**CAMILO CASTRO - CAMILO@NINJAS.CL**

# TEMARIO

## TEMAS PRINCIPALES

- ¿Qué es el Technical Weight?
- Estrategias
- Ejemplos

# **TECHNICAL DEBT**

# TECHNICAL DEBT

ACUÑADO POR WARD CUNNINGHAM EN 1992 (1)

**Javier Garzas:** “La deuda técnica es el coste y los intereses a pagar por hacer mal las cosas. El sobre esfuerzo a pagar para mantener un producto software mal hecho, y lo que conlleva, como el coste de la mala imagen frente a los clientes, etc.”. (2)



[1] <http://c2.com/doc/oopsla92.html>

[2] <https://www.javiergarzas.com/2012/11/deuda-tecnica-2.html>

# **TECHNICAL WEIGHT**

# TECHNICAL WEIGHT

ACUÑADO POR BART WRONSKI EN 2016 (1)

**El peso técnico es una característica de una solución que la puede tornar costosa y "pesada" de mantener a mediano y largo plazo. Incluso en ambientes limpios y apropiadamente diseñados (con baja deuda técnica).**



[1] <https://bartwronski.com/2016/06/26/technical-weight/>

# ANALOGÍAS

## FERRARI

- Costo circulación elevado (impuesto al lujo).
- Costo de reparación y repuestos elevado (Difíciles de encontrar).



# ANALOGÍAS

## EQUIPAJE

- Si realizas un viaje. ¿Cuánto peso puedes llevar en tu mochila?.
- ¿Usarás todo lo guardado en tu equipaje?.
- Puedes deshacerte del peso extra en el camino, pero ¿Se justifica la inversión inicial?.



# T. WEIGHT VS T. DEBT

## RELACIÓN

- Un sistema con alto peso técnico puede tener alta deuda técnica.
- Un sistema con baja deuda técnica puede tener alto peso técnico.
- Toda decisión técnica tiene un peso.



# ¿POR QUÉ ES BUENO CONTROLARLO?

- Mano de obra necesaria para mantener el sistema.
- Dificultad de encontrar nuevos devs.
- Costos de reparación de bugs.
- Refactorización complicada.
- Extensión complicada.
- Prejuicios:
  - Sesgo de confirmación
  - Aumento del compromiso
  - Trampa del progreso
  - Aversión a la pérdida



# ¿POR QUÉ ES BUENO CONTROLARLO?

- Douglas Hubbard, en su libro “How to measure anything”, define el riesgo como “un estado de incertidumbre donde algunas de las posibilidades pueden implicar pérdidas, catástrofes u otros resultados no deseados”.



# EJEMPLO

[HTTPS://WWW.NICOLAS-SCHURMANN.COM/KLEISLI/](https://www.nicolas-schurmann.com/kleisli/)

```
1 const composeMap = (...ms) =>
2   ms.reduce((f, g) => x => g(x).map(f))
3
4 const comp = composeMap(
5   funcion1,
6   funcion2,
7 )
8
9 const comp2 = x => x.map(funcion1).map(funcion2)
10
11 comp === comp2 // en este caso, el resultado es el mismo
```



# EJEMPLO

**¿POR QUÉ NO SIEMPRE ES BUENO USAR ALGO QUE CLARAMENTE ES SUPERIOR?**

- **Las soluciones adoptadas dependerán de condiciones de cada proyecto. Lo que pudo servir para uno, puede ser muy “pesado” de utilizar en otro.**
- **Se debe considerar no solo la implementación inicial, también los costos de mantenimiento y extensión.**



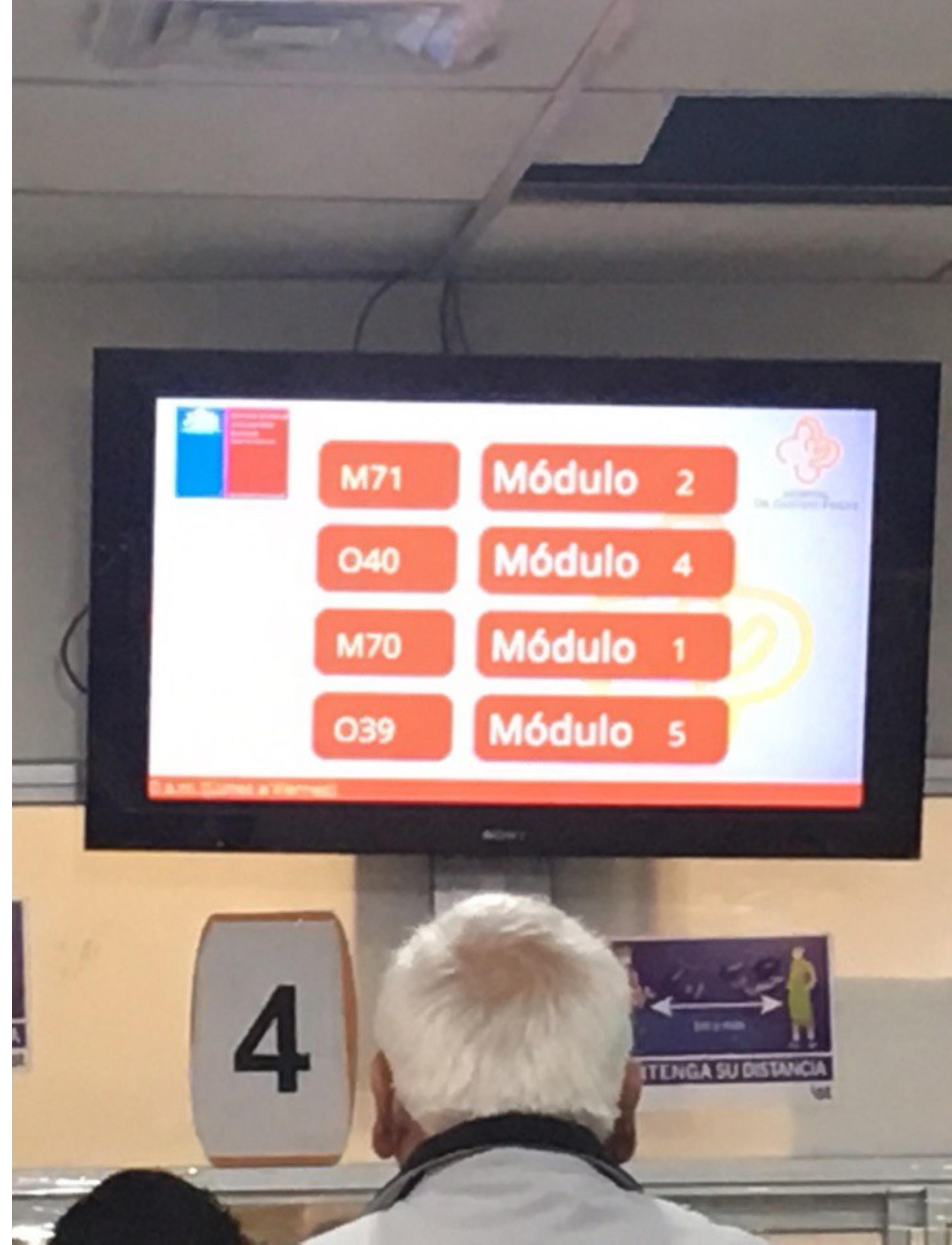
# **EJEMPLO: SISTEMA DE TURNOS**

<https://github.com/ninjascl/screensharer>

# SISTEMA DE TURNOS

## PROBLEMAS

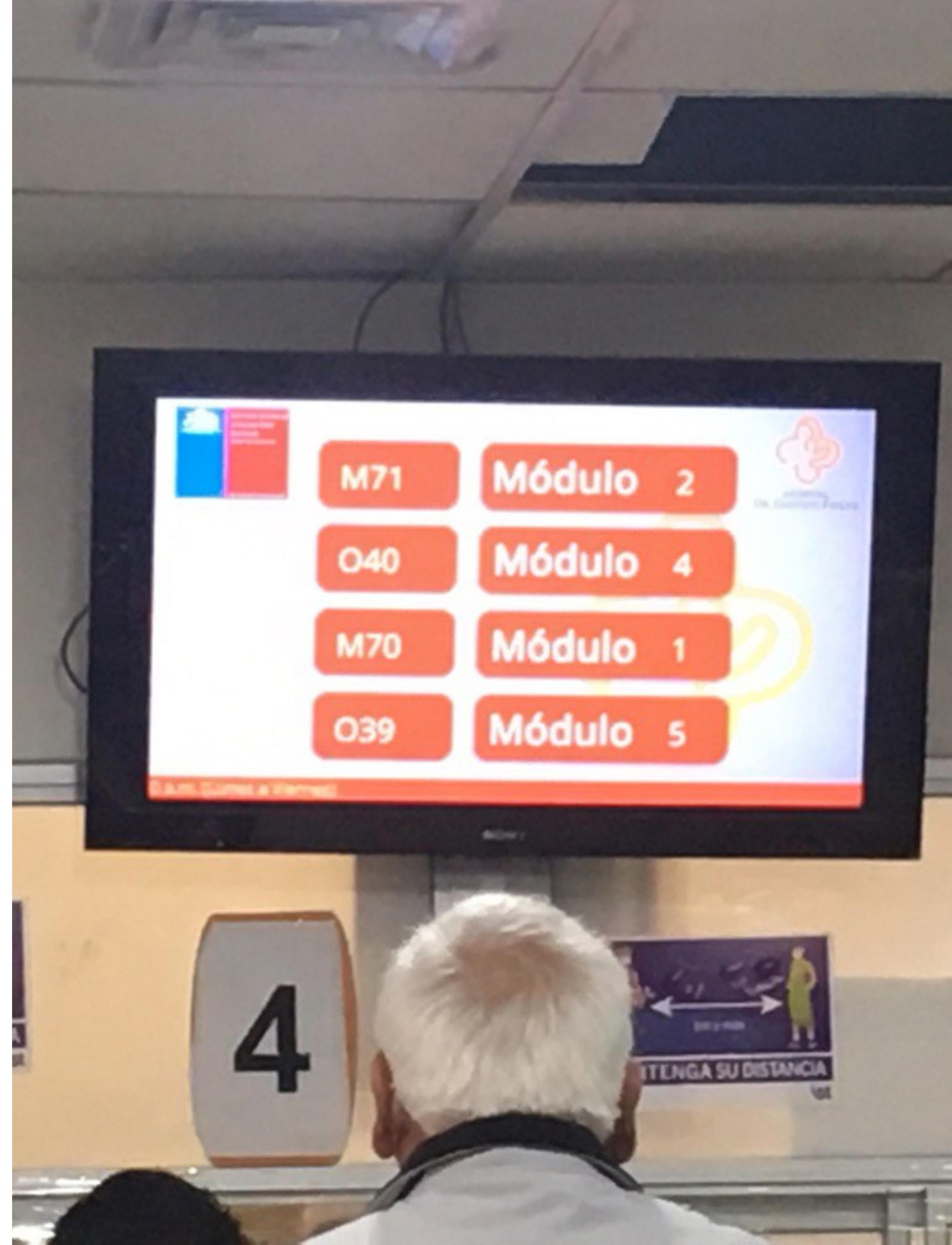
- **Información offline.**
- **Necesidad de asistir físicamente y esperar por largos períodos.**
- **Riesgo de contagio de enfermedades.**



# SISTEMA DE TURNOS

## DESAFÍO FOCAL

- Personas obligadas a esperar en el sitio para saber su turno.



# SISTEMA DE TURNOS

## POSIBLES SOLUCIONES (CON POCO PESO TÉCNICO)

- Compartir pantalla de turnos en página web.
- Persona encargada de llamar e informar por teléfono.



# SISTEMA DE TURNOS

## COMPARTIR PANTALLA EN PÁGINA WEB

- Se saca un screenshot cada x segundos del computador usado en la sala de espera.
- El archivo se sube a un servidor web o almacenamiento externo (Firebase, S3, FTP).
- La página web se refresca automáticamente cada x segundos mostrando el último screenshot disponible.



# SISTEMA DE TURNOS

PERSONA ENCARGADA DE LLAMAR E INFORMAR POR TELÉFONO.

- Persona encargada puede ser contactada por SMS/Whatsapp/Llamada.
- Saca turno manualmente con los datos del paciente.
- Llama o envía mensaje al paciente cuando queden 20 números para llegar al indicado.
- Pensado para personas sin acceso a internet o dificultades técnicas (abuelitos).



# SISTEMA DE TURNOS

## RESULTADO DE LAS SOLUCIONES

- La información está disponible online.
- Se reduce el tiempo necesario de estar presentes. (Solamente se requiere un tiempo mínimo para realizar el trámite mismo).
- Al requerir menor tiempo presencial, se reduce el riesgo de enfermarse (menor aglomeración de gente).



# ¿POR QUÉ SE GENERA?

- Problema difícil
- Sobre dimensionar un problema
- Poca acotación del alcance
- Funcionalidad Futura
- No dispuesto a comprometerse
- Poca experiencia



# ¿POR QUÉ SE GENERA?

- Dogmatismo técnico
- Diversión
- Ser “ingenioso”
- Progreso de carrera individual
- Progreso de carrera organizacional

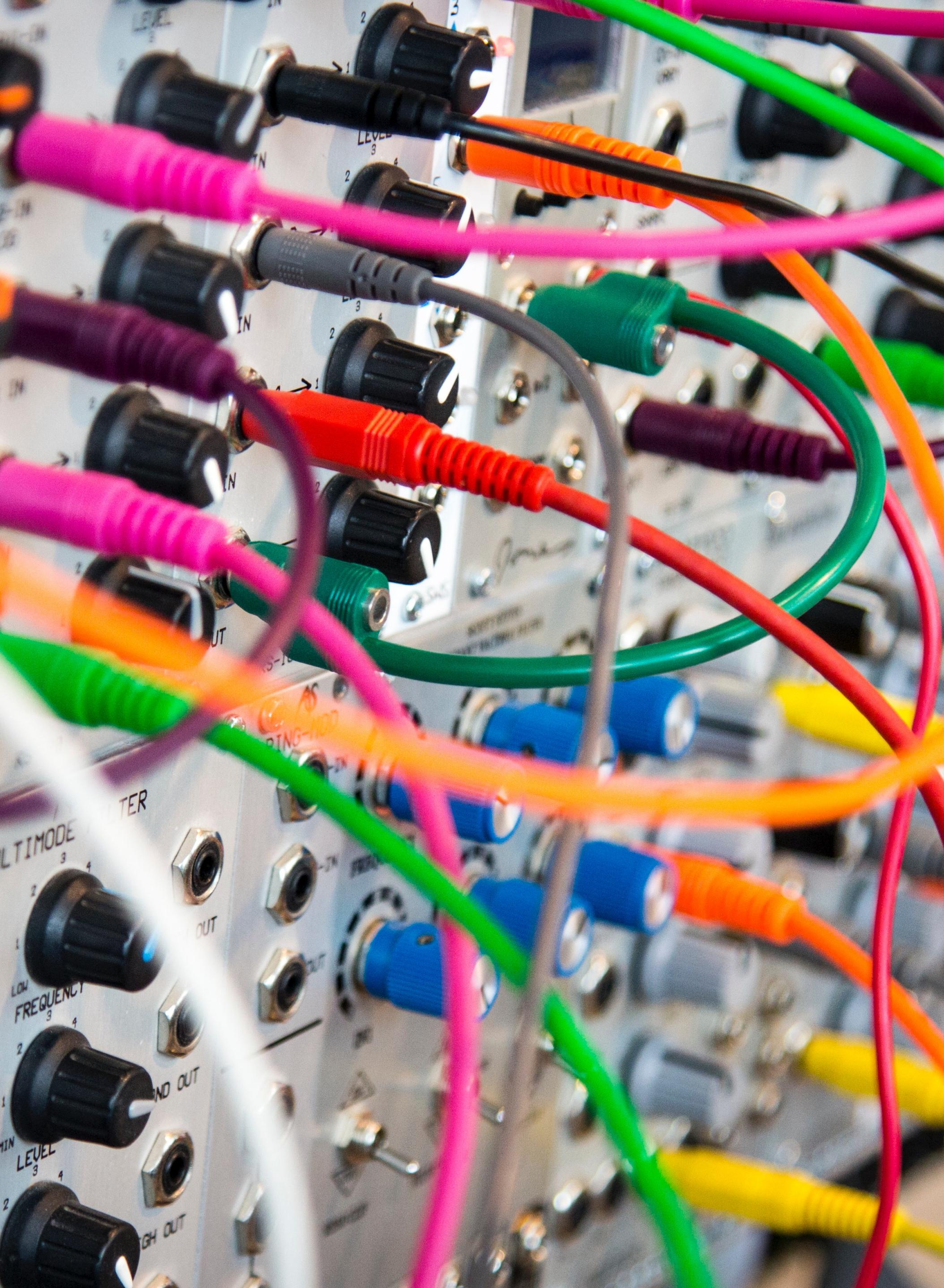


# **ESTRATEGIAS**

# DEFINIR PESO (COMPLEJIDAD)

DONALD E. KNUTH: [HTTPS://DL.ACM.ORG/DOI/10.1145/1008328.1008329](https://dl.acm.org/doi/10.1145/1008328.1008329)

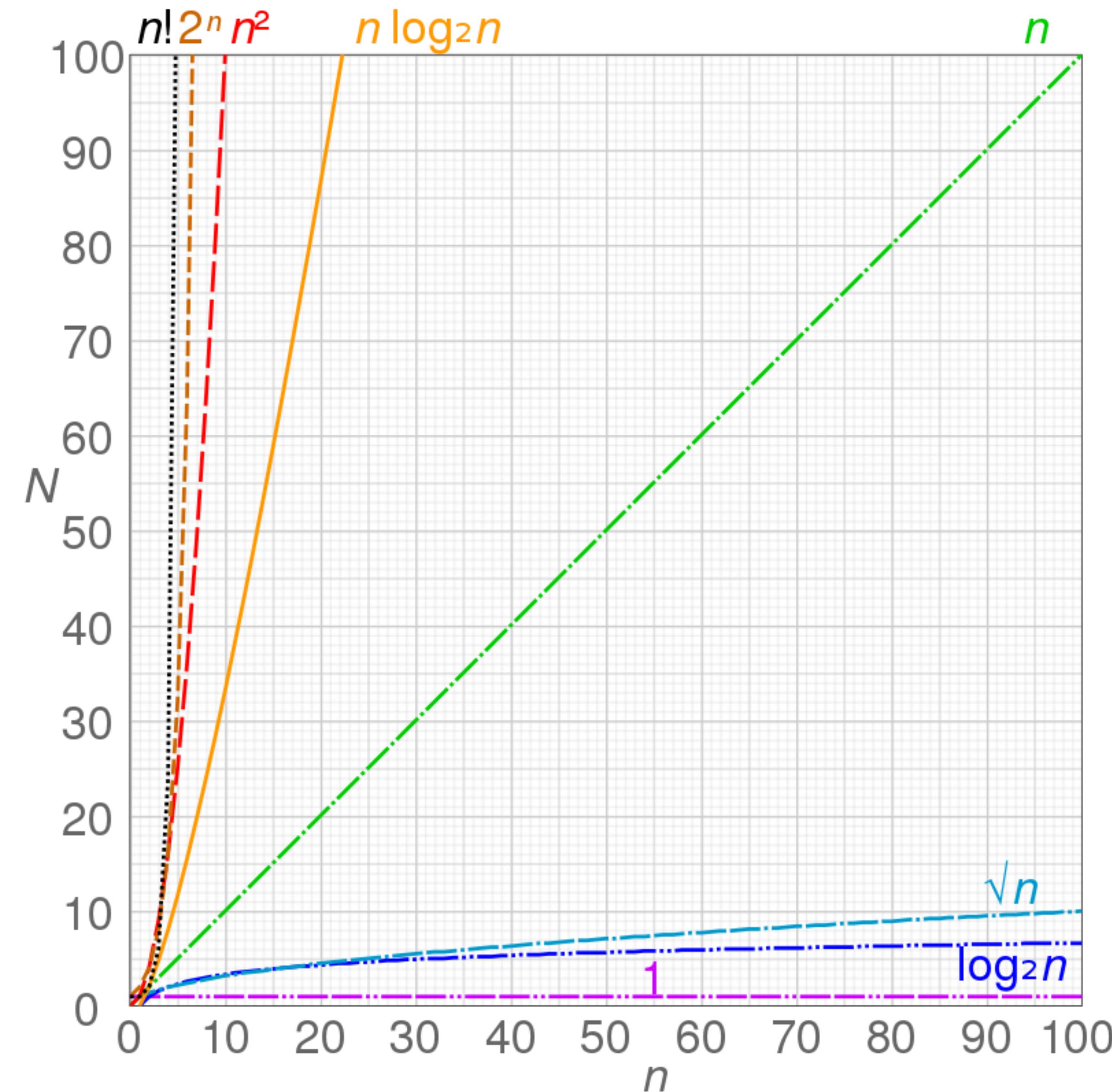
- Big Omicron  $O( f(n) )$  : Como máximo  $f(n)$ .
- Big Omega  $\Omega( f(n) )$  : Al menos  $f(n)$ .
- Big Theta  $\Theta( f(n) )$  : Exactamente  $f(n)$ .
- Mide complejidad de ejecución (runtime).
- Mide complejidad de espacio (space).
- Podría medir complejidad técnica (weight).
- ¿Cuál es tu  $O(1)$ ?



# DEFINIR PESO (COMPLEJIDAD)

[HTTPS://EN.WIKIPEDIA.ORG/WIKI/BIG\\_O\\_NOTATION](https://en.wikipedia.org/wiki/BIG_O_NOTATION)

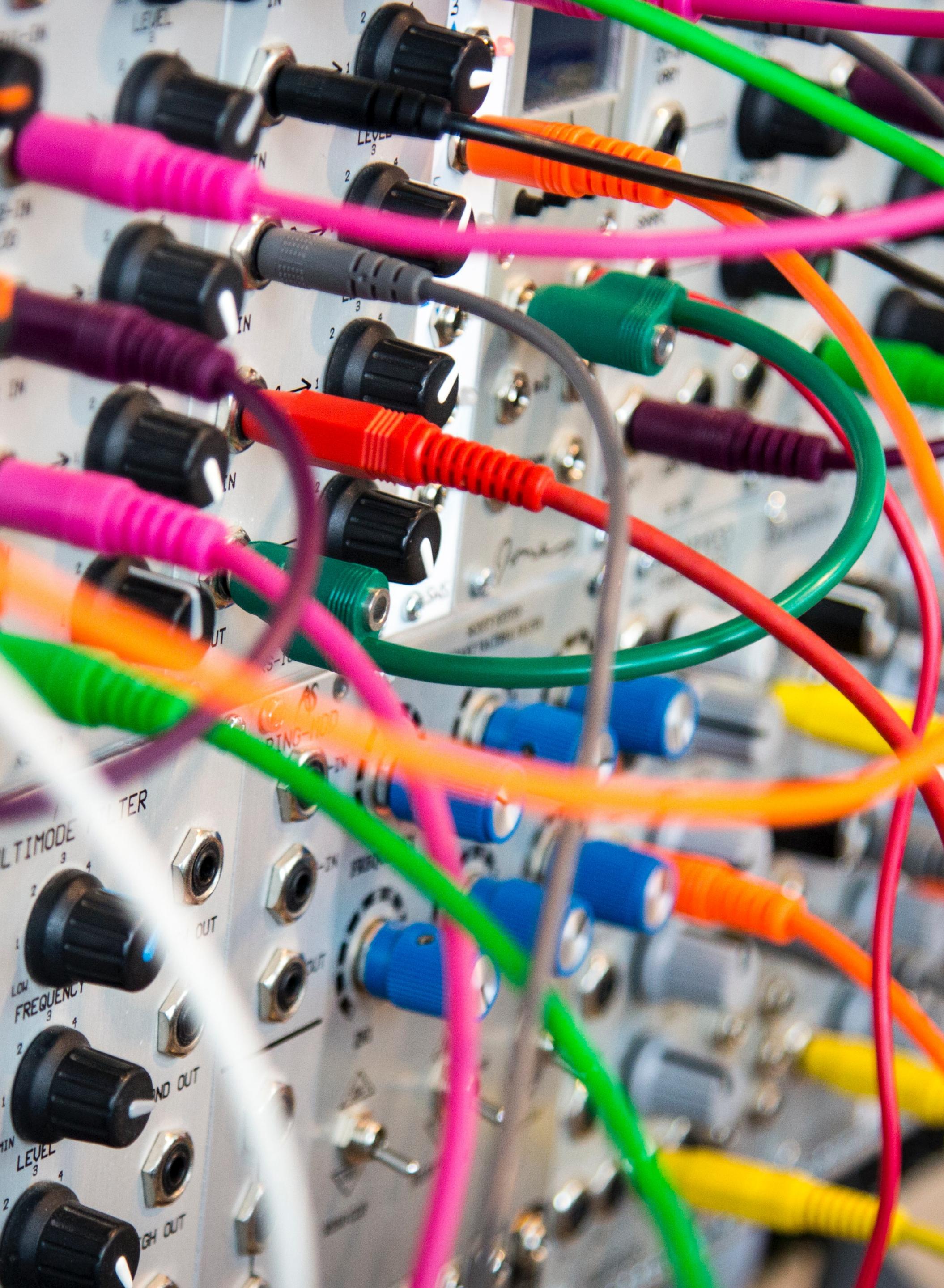
- $O(1)$  = El mejor escenario.
- $O(\log^2 n)$
- $O(\sqrt{n})$
- $O(n)$  = Escenario promedio.
- $O(n \log^2 n)$
- $O(n^2)$
- $O(2^n)$
- $O(n!)$  = El peor escenario.



# DEFINIR PESO (COMPLEJIDAD)

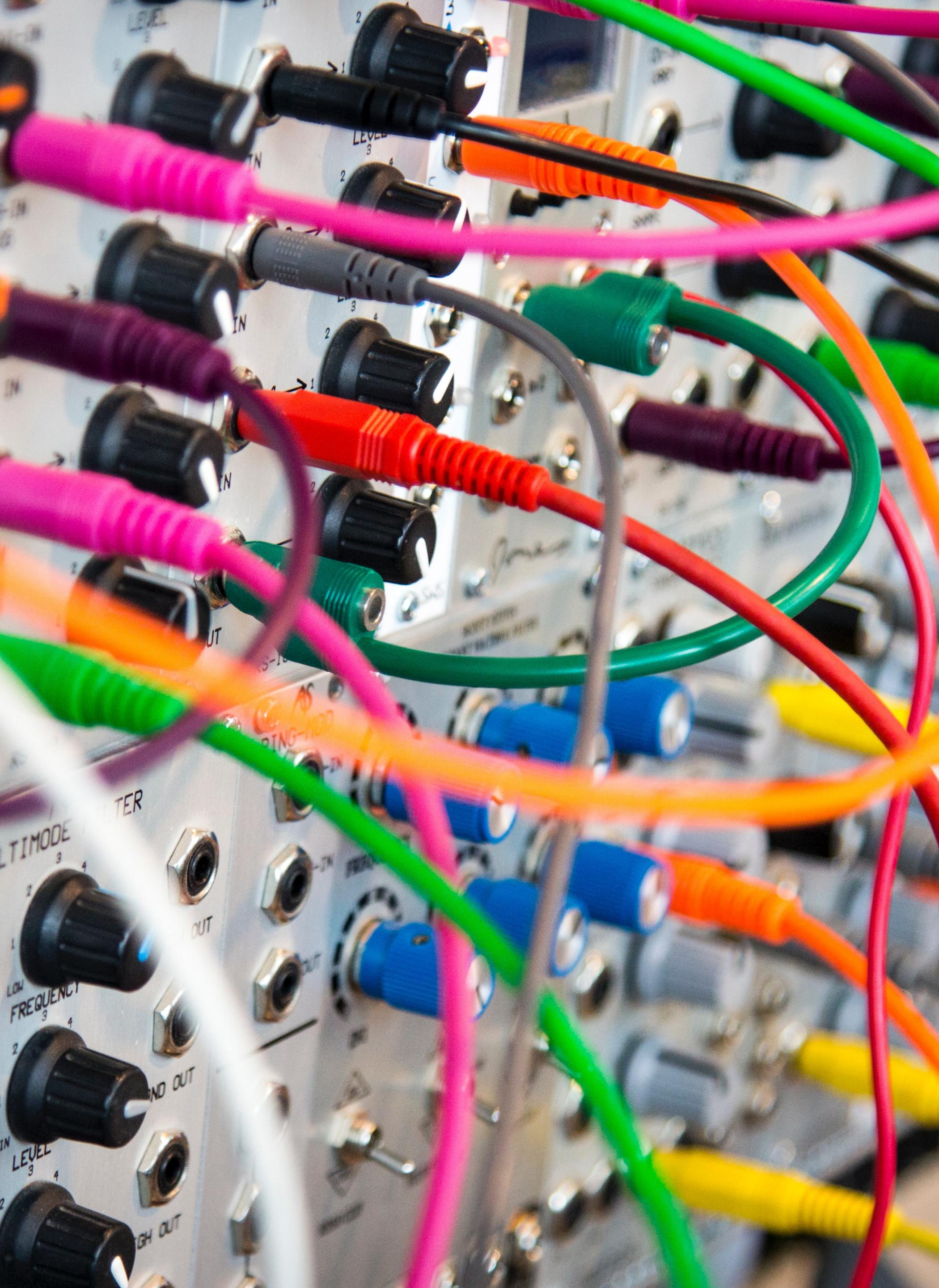
[HTTPS://ES.WIKIPEDIA.ORG/WIKI/ENTROP%C3%ADA](https://es.wikipedia.org/wiki/Entrop%C3%ADa)

- Considerar la Entropía. (Nivel de libertad).  
Mayor orden = menor libertad.
- La cantidad de entropía de un sistema es proporcional al logaritmo natural del número de microestados posibles ( $S = k \ln \Omega$ ).
- El tiempo pasa y la entropía crece hasta alcanzar el punto de máxima entropía del sistema.



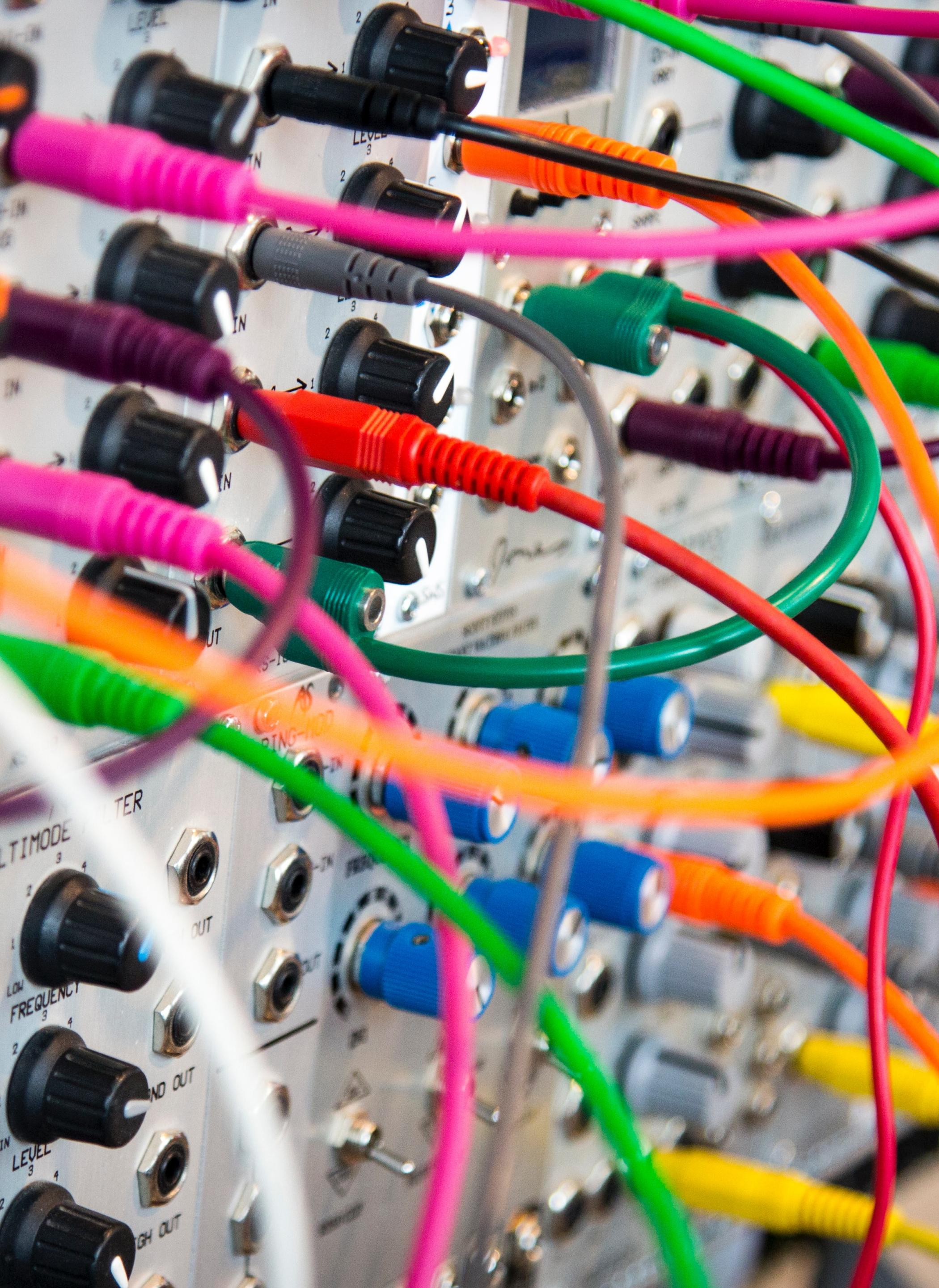
# DEFINIR PESO (COMPLEJIDAD)

- ¿Cuánto peso técnico tienes?.
- ¿Por cuánto tiempo lo piensas sostener?.
- ¿Podrás continuar sosteniéndolo si las personas se van del equipo?.
- ¿Realmente se aprovechan todas las características del producto?.



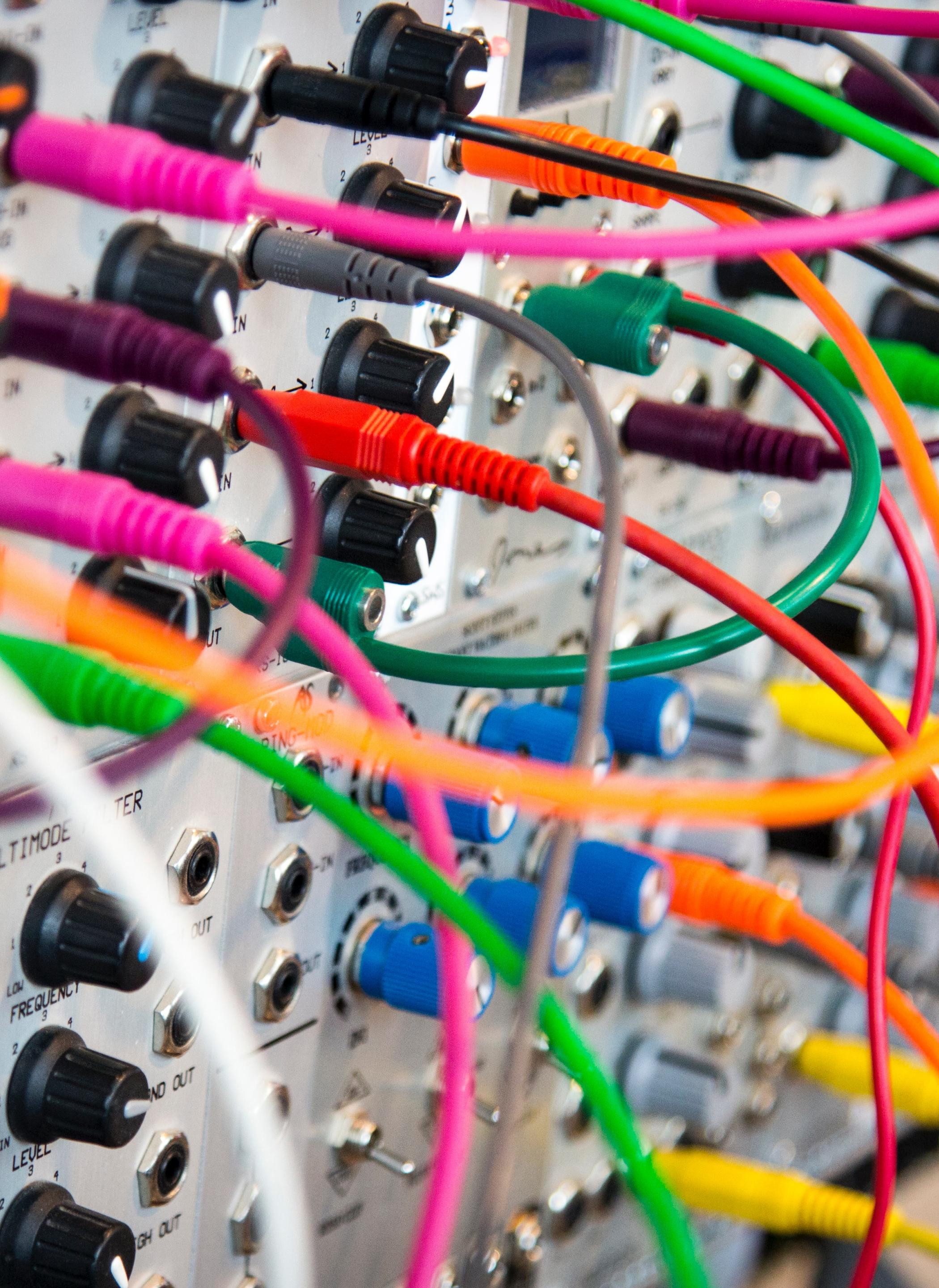
# DEFINIR PESO (COMPLEJIDAD)

- ¿Tratas de ser “ingenioso” e impresionar?
- ¿Qué desventajas hay en usar soluciones más ligeras?. ¿Existe algún prejuicio?.
- ¿Qué alternativas existen?.



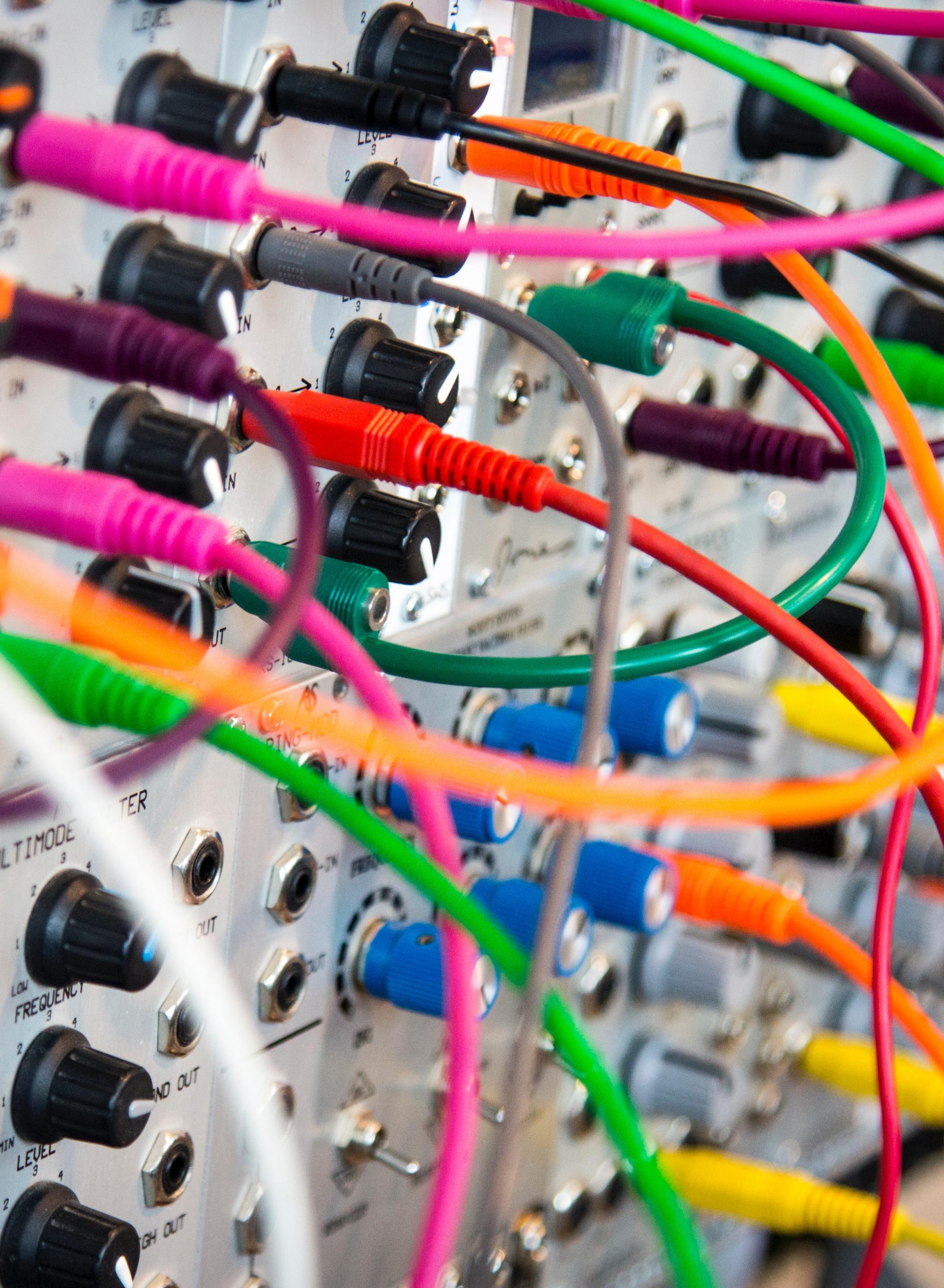
# DEFINIR PESO (COMPLEJIDAD)

- Medir apoyándose en personas con experiencia y comparar estimaciones.
- Medir la cantidad de personas disponibles en el mercado laboral. ¿Qué tan fácil o costoso es encontrar personas que deseen trabajar con el stack tecnológico seleccionado?, ¿Qué tan fácil es su capacitación?.



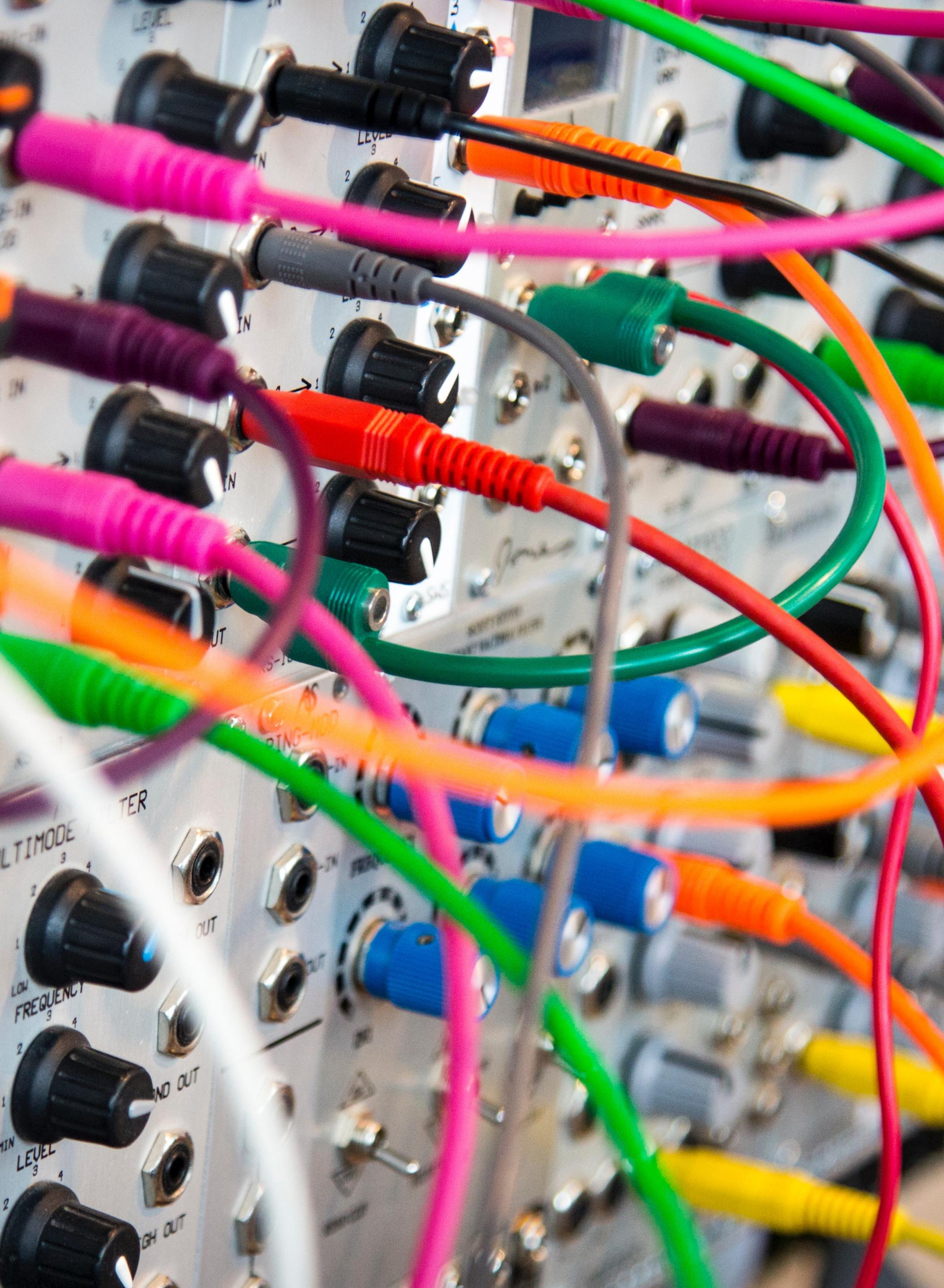
# DEFINIR PESO (COMPLEJIDAD)

- Distintas organizaciones pueden soportar distinto peso.
- Lo que funcionó para una organización pequeña, puede no servir para una organización más grande y viceversa.
- Una ecuación general para definir peso no es posible debido a la naturaleza cambiante de proyectos y organizaciones. Solo es posible una aproximación.



# DEFINIR PESO (COMPLEJIDAD)

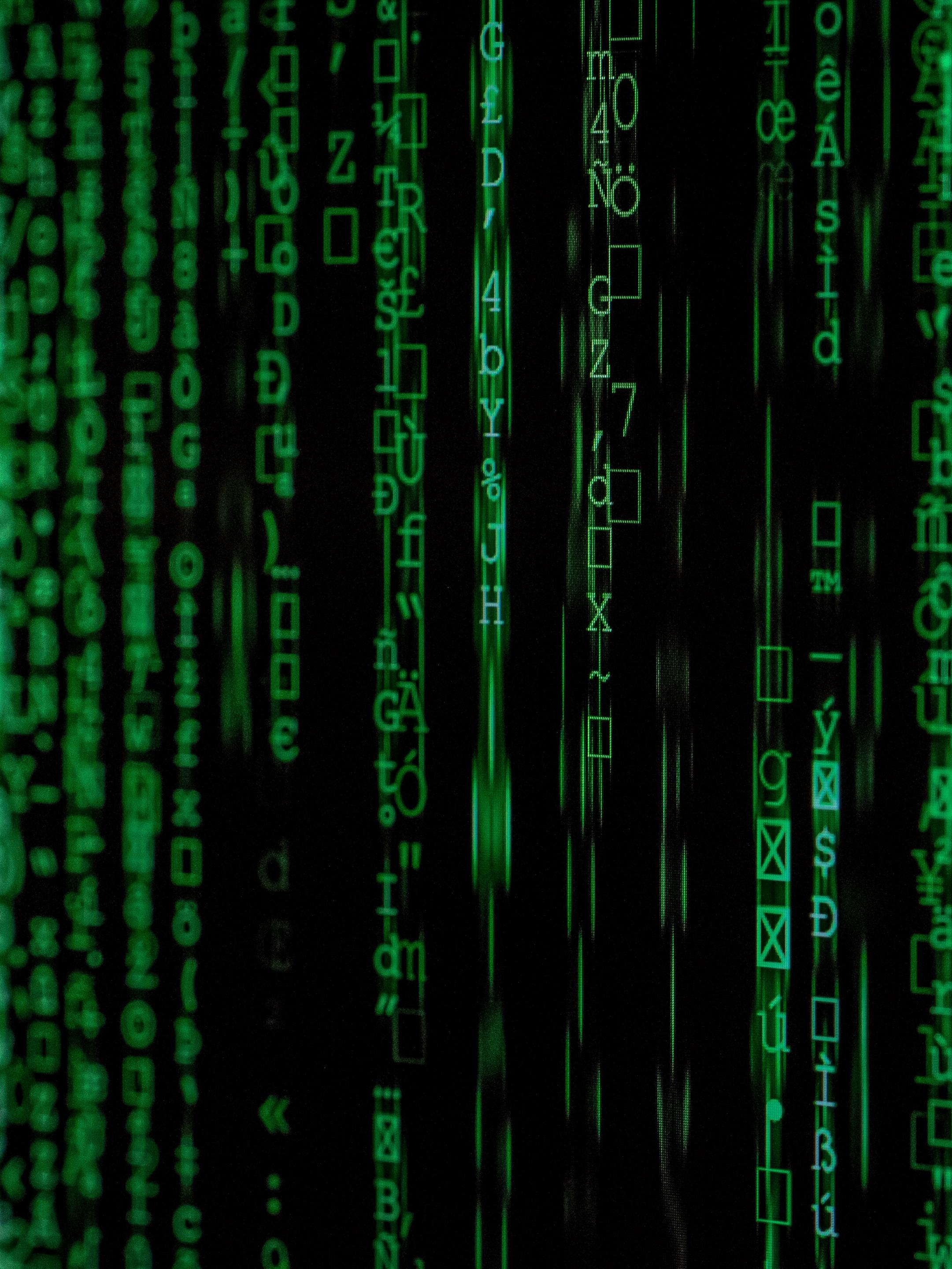
- Evaluar sistemas “legacy”. Mejorarlos, simplificarlos o desecharlos.
- Todo cambio debe ser estudiado, gradual y aprobado por el equipo involucrado y Stakeholders.
- «Apenas se puede negar que el objetivo supremo de toda teoría es hacer los elementos básicos tan simples y tan poco numerosos como sea posible, sin renunciar a la representación adecuada de un simple dato» - Albert Einstein



# USAR TECNOLOGÍA CONOCIDA

DAN MCKINLEY: [HTTP://BORINGTECHNOLOGY.CLUB/](http://BORINGTECHNOLOGY.CLUB/)

- Permite conocer fortalezas y debilidades de cada solución.
- Mayor cantidad de acervo documental.
- Usar pruebas de concepto, prototipos y MVP para tecnologías más experimentales.
- Usar tecnologías conocidas y probadas para proyectos que requieran mayor control de riesgos. (Se puede usar tecnologías más nuevas, siempre que se conozcan bien).
- Buscar utilizar la menor cantidad de elementos posibles.  
¿Cómo puedo lograrlo utilizando lo que ya tengo disponible, sin agregar nuevos componentes?.



# ENCONTRAR EL DESAFÍO FOCAL

HBR: [HTTPS://HBR.ORG/2015/12/FIND-INNOVATION-WHERE-YOU-LEAST-EXPECT-IT](https://hbr.org/2015/12/find-innovation-where-you-least-expect-it)

- ¿Dónde está el problema raíz?.
- ¿Puede un problema ser usado como su propia solución?.
- Brainstorming “mudo”. Utilizando tarjetas y texto en vez de voz. Da más oportunidades a personas tímidas.
- Root Cause Analysis.



# CRÉDITOS

## FOTOGRAFÍAS USADAS

<https://unsplash.com/photos/rX12B5uX7QM> Photo by Ian Espinosa on Unsplash

<https://unsplash.com/photos/Yuv-iwByVRQ> Photo by Victor Freitas on Unsplash

<https://unsplash.com/photos/SPbcqTVoYqE> Photo by Matt Lamers on Unsplash

<https://unsplash.com/photos/oNDRCGrqaYc> Photo by Karim MANJRA on Unsplash

<https://unsplash.com/photos/1F4MukO0UNg> Photo by Glenn Carstens-Peters on Unsplash

<https://unsplash.com/photos/3y1zF4hIPCg> Photo by Hans-Peter Gauster on Unsplash

<https://unsplash.com/photos/TsVN31Dzyv4> Photo by Cyril Saulnier on Unsplash

<https://unsplash.com/photos/I090uFWoPal> Photo by John Barkiple on Unsplash

[https://unsplash.com/photos/DX3\\_dXuHVI8](https://unsplash.com/photos/DX3_dXuHVI8) Photo by Immo Wegmann on Unsplash

<https://unsplash.com/photos/m2TU2gfqSeE> Photo by You X Ventures on Unsplash