Panorama 2025-2030 y fundamentos de Inteligencia Artificial

De modelos fundacionales a agentes y equipos

Aplicaciones, Modelos y Infraestructura

Temario

- Tipos de aprendizaje: supervisado, no supervisado, autosupervisado, refuerzo (+ few-shot)
- Representaciones: embeddings y espacio latente (la "API" de similitud)
- Pipeline típico de ML: datos -> modelo -> evaluación -> despliegue (con monitoreo y drift)
- Modelos fundacionales y reutilización multi-tarea
- Tendencias 2025-2030: escalamiento de modelos y cómputo, IA como infraestructura

La pila de la IA: tres capas (dónde se juega el valor)

Aplicaciones
Prompts + contexto
RAG / herramientas
Evaluación & guardrails
UX (web/chat/voz)

Modelos
Pretraining / finetuning
Ingeniería de datos
Post-training
Optimización de inferencia

Infraestructura
Serving + clusters GPU
Datos + observabilidad
Seguridad
Coste y escalabilidad

Capa aplicaciones: del prompting a flujos agentivos

- Personalización práctica: instrucciones, formato, criterios y rol del asistente
- Contexto: RAG, memoria de sesión, herramientas (BD, docs, web, APIs)
- Flujos agentivos: planificar -> actuar -> verificar (rutinas repetibles)
- Evaluación crítica por salidas open-ended: tests, rúbricas, datasets dorados
- Guardrails y responsabilidad: límites, permisos, auditoría, revisión humana

Capa modelos: del pretraining a la adaptación eficiente

- Pretraining (autosupervisado) para capacidades generales -> representaciones (embeddings) reutilizables
- Adaptación eficiente: prompting/few-shot, RAG, fine-tuning, adapters (LoRA)
- Evaluación: no solo accuracy; también robustez, sesgos, seguridad y monitoreo en producción
- Ajuste de datos: curación, deduplicación, tokenización, anotación (data-centric AI)

IA como infraestructura (y no solo software)

- Serving: despliegue de modelos y gestión de cómputo (GPU/TPU, colas, batch/streaming)
- Datos: calidad, versionado, linaje; feature stores / vector stores (si aplica)
- Observabilidad: trazas, logs, métricas (calidad, costo, seguridad, drift)
- Requisitos transversales: escalabilidad, seguridad, privacidad y coste
- Restricciones reales: energía, refrigeración, redes, disponibilidad de hardware

Tipos de aprendizaje: qué señal optimiza cada uno

- **SUPERVISADO** - señal: error vs. etiqueta (x,y)
  Ejemplo.: spam, diagnóstico por imagen, pronóstico de demanda
  Few-shot: mismo objetivo con pocos (x,y); apoyar con pretraining/embeddings

- **NO SUPERVISADO** - señal: estructura en x (distancias/densidad/reconstrucción)
  Ejemplo: clustering/segmentación, anomalías en logs, reducción de dimensión

AUTOSUPERVISADO - señal creada (pretext): predecir partes faltantes o similitud
Ejemplo: next-token / masked LM, contrastivo (SimCLR), autoencoders
Produce representaciones (embeddings) reutilizables

- **REFUERZO (RL)** - señal: recompensa acumulada
  Ejemplo: robótica, recomendación secuencial, scheduling; RLHF en LLMs

Regla práctica: generalización multi-tarea -> autosupervisado + adaptación; control secuencial -> RL o híbridos.

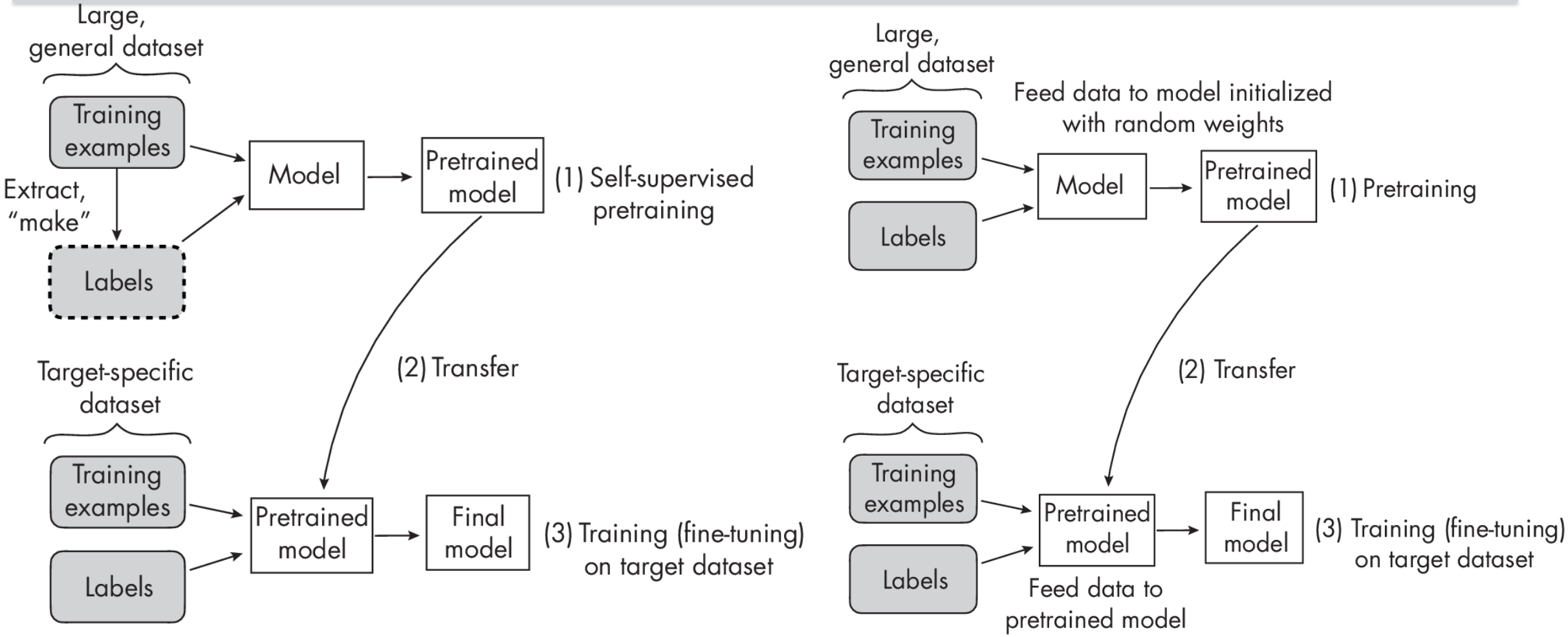
Transfer learning vs. auto-supervisado

- Transfer learning: preentrenas con datos etiquetados y luego ajustas (fine-tuning) en la tarea objetivo.
- Auto-supervisado: preentrenas con datos no etiquetados derivando "etiquetas" del propio dato (tareas pretexto).

Cuándo conviene: redes grandes + pocas etiquetas -> auto-supervisado; etiquetas masivas disponibles -> transferencia.

Nota práctica: árboles/GBDT no suelen beneficiarse igual (no hay representaciones reutilizables aprendidas por capas).

Casos


Large,
general dataset
Training examples
Extract, "make"
Labels
Model
Pretrained model
(1) Self-supervised pretraining
(2) Transfer
Target-specific dataset
Training examples
Labels
Pretrained model
Final model
(3) Training (fine-tuning) on target dataset
Large,
general dataset
Training examples
Labels
Feed data to model initialized with random weights
Model
Pretrained model
(1) Pretraining
(2) Transfer
Target-specific dataset
Training examples
Labels
Pretrained model
Feed data to pretrained model
Final model
(3) Training (fine-tuning) on target dataset

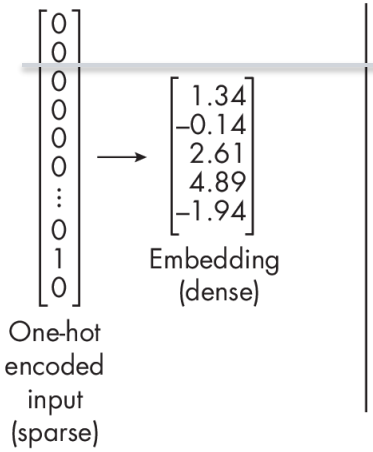
Embeddings, espacio latente y representaciones

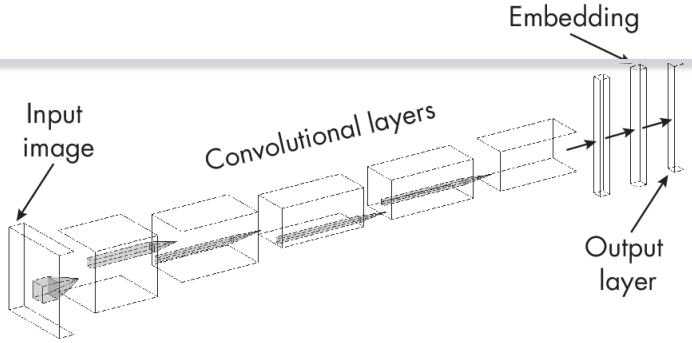
Embeddings: mapean datos (a menudo de alta dimensión) a vectores densos; preservan similitud/distancia.

Espacio latente: espacio de características donde "viven" esas representaciones (p.ej., cuello de botella de un autoencoder).

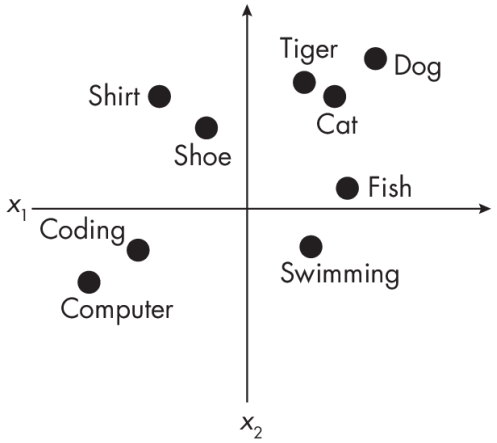
Representación: cualquier codificación útil (incluye one-hot, features manuales o capas intermedias).

>Por qué importa (2025-2030): clustering y búsqueda por similitud, few-shot vía vecinos/prototipos, RAG y evaluación basada en embeddings.

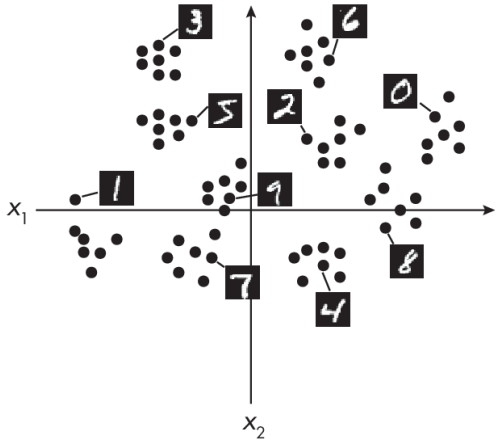




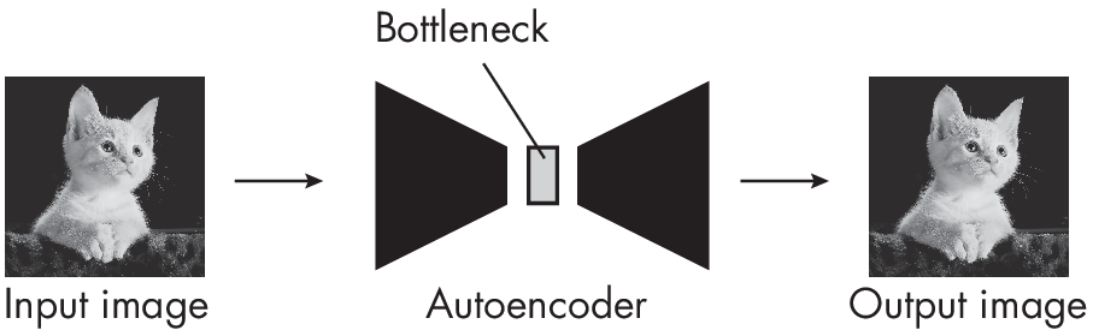
Embeddings learned by a convolutional network



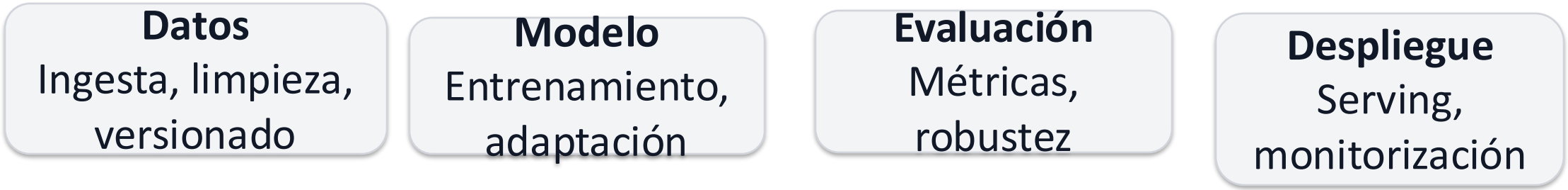
2D word embeddings



2D handwritten digit embeddings



Pipeline típico de ML



El pipeline es un bucle: en producción aparece drift/shift -> re-entrenamiento y recalibración. 2025-2030: además de métricas, importa el sistema: datos versionados, auditoría, permisos y seguridad.

Punto clave: 'data-centric AI' y monitoreo continuo suelen dar más ROI que cambiar arquitectura.

Evaluación en IA generativa: tests + guardrails

- **Problema:** salidas *open-ended* -> una sola métrica no basta (accuracy ≠ calidad) y hay riesgo de alucinación/PII/inyección.
- **Base:** *golden datasets* versionados por tarea + **regresión** (pass/fail por checks).
- **Criterio: rúbricas** (factualidad, cobertura, claridad, formato, seguridad) + revisión humana en casos críticos.
- **Estrés: pruebas adversariales** (prompts maliciosos, ambigüedad, edge cases, datos ruidosos) para medir robustez.
- **Control: guardrails** (allowlist de herramientas, políticas, verificación de esquema/PII/claims vs fuentes, logs y auditoría) + patrón **generador -> verificador** (o multiagente).

Modelos fundacionales: qué son y por qué importan

- **Qué los hace "fundacionales"**
  - Preentrenamiento masivo (texto/código/imágenes) -> **capacidades generales**
  - **Representaciones reutilizables** + habilidades emergentes multi-tarea
- **Reutilización / adaptación**
  - **Prompting**: instrucciones + ejemplos + formato
  - **RAG**: contexto con fuentes (docs/BD) -> mejor precisión y trazabilidad
  - **Fine-tuning / adapters (LoRA)**: especialización barata sin reentrenar todo
  - **Tool use**: orquestación de buscadores, código y APIs
- **Ventaja**
  - "Pones el modelo en el contexto correcto" -> **más valor con menor costo**
  - Un modelo sirve para muchas tareas (resumen, extracción, clasificación, soporte, etc.)

Riesgos y cómo se controlan (evaluación + guardrails)

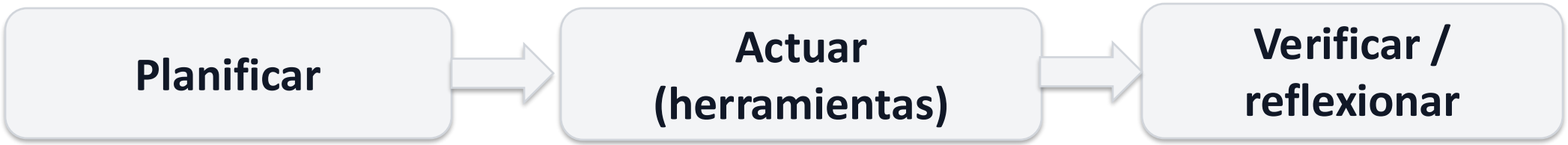
- **Riesgo: "caja negra"**
  - Alucinación, sesgo, fuga de datos, prompt injection, exceso de confianza
  - Impacto: errores silenciosos + decisiones no auditables
- **Respuesta técnica mínima viable**
  - **Evaluación**: suite de casos + checks (JSON válido, límites, PII, factualidad, consistencia)
  - **Guardrails**: políticas, validadores, límites de herramientas, sandboxing, rate limits
  - **Monitoreo**: logs, métricas, drift, incidentes y retroalimentación continua
- **Responsabilidad (gobernanza)**
  - Trazabilidad de fuentes (RAG), revisión humana en decisiones críticas, auditoría

De modelo a agente: planificar -> actuar -> verificar

- **Modelo (LLM):** genera texto/decisiones **dado un prompt**. No ejecuta acciones por sí mismo.
- **Agente:** un sistema que usa un modelo + **objetivo** + **memoria/estado** + **herramientas** para **tomar acciones** en un entorno y cerrar un ciclo hasta lograr una meta.

Bucle operativo del agente

- **Planificar -> Actuar -> Verificar**
  - *Planificar:* descomponer objetivo en pasos (tareas/subtareas)
  - *Actuar:* llamar herramientas (APIs) y producir artefactos
  - *Verificar:* validar resultados (tests, reglas, fuentes) y corregir si falla



Herramientas típicas

- Búsqueda web, **BD**/SQL, hojas de cálculo, correo, repositorios (Git), tickets, ejecución de código, RAG sobre documentos.

Controles (seguridad y gobernanza)

- **Permisos mínimos** (least privilege), **sandbox**, límites de acción (rate limits / scope), confirmaciones humanas en acciones sensibles, **registro y auditoría** de decisiones y fuentes.

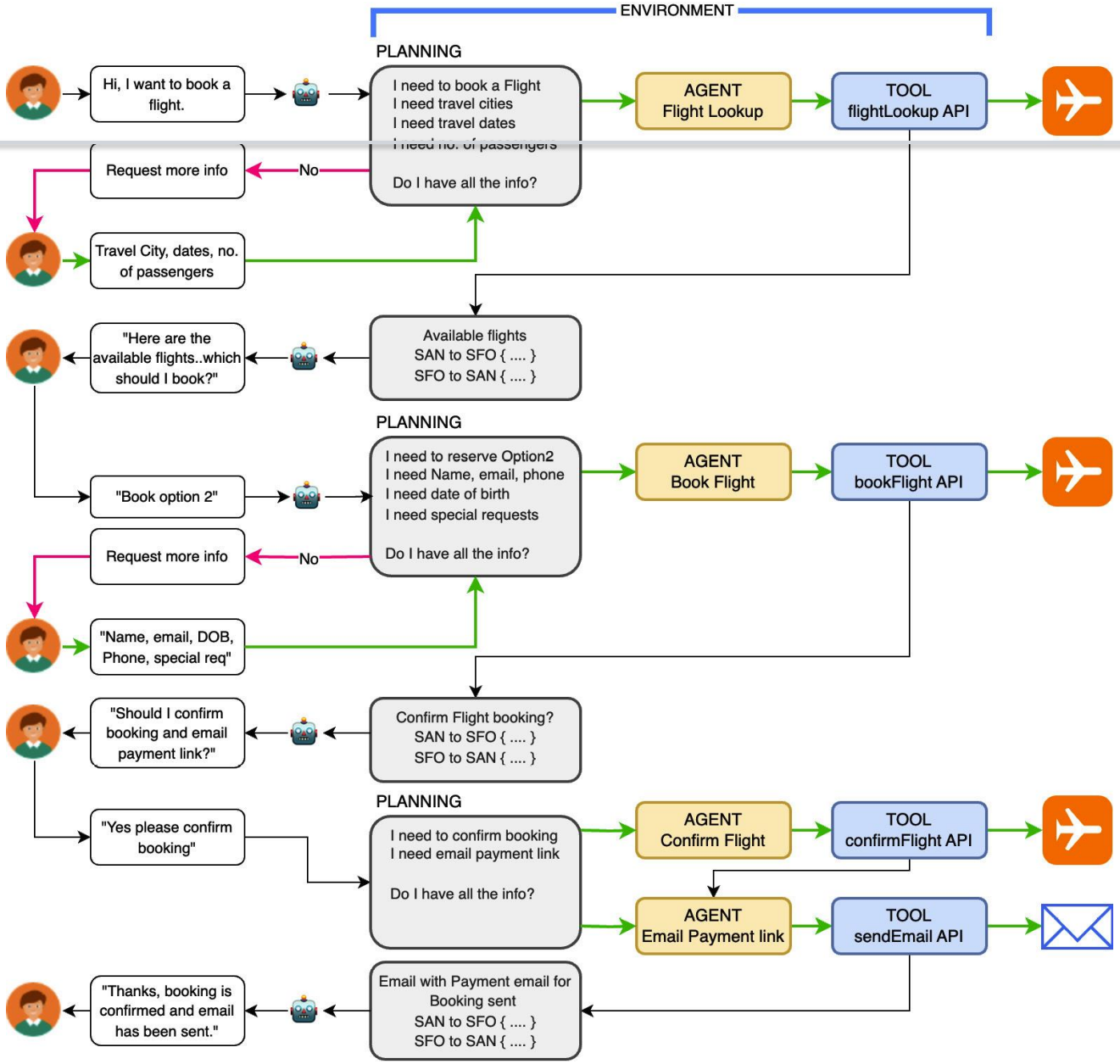
Principio clave

- **Humano "piloto a cargo"**: supervisa, aprueba acciones críticas y asume **rendición de cuentas** (el agente acelera, no reemplaza responsabilidad).

Multiagentes: "Divide y vencerás" para robustez

- **Concepto (multiagente):** equipo de agentes con **roles complementarios** para dividir trabajo y reducir errores.
  - Roles típicos: **analista**, **redactor**, **crítico**, **verificador**, **auditor**
  - Variante útil: **Red Team / Blue Team** (proponer vs atacar) + "juez" con rúbrica
- **Beneficios:**
  - **Mayor cobertura** (más perspectivas y edge cases)
  - **Menos errores** (detección temprana de inconsistencias/alucinaciones)
  - **Mejores decisiones** (argumentos + evidencia + evaluación)
- **Riesgos / trade-offs:**
  - **Bucles** y "discusiones infinitas"
  - **Contradicciones** entre agentes (falta de criterio de decisión)
  - **Costos/latencia** más altos -> requiere **orquestación** y **condiciones de parada**
- **Patrón recomendado (con evidencia):**
  - **Generador -> Verificador -> Integrador**
  - Verificador exige **evidencia** (RAG/citas/logs/tests) y el Integrador consolida + decide según rúbrica

Un ejemplo


ENVIRONMENT
PLANNING
Hi, I want to book a flight.
I need to book a Flight
I need travel cities
I need travel dates
I need no. of passengers
Do I have all the info?
AGENT
Flight Lookup
TOOL
flightLookup API
Request more info
No
Travel City, dates, no. of passengers
"Here are the available flights..which should I book?"
Available flights
SAN to SFO { .... }
SFO to SAN { .... }
PLANNING
"Book option 2"
I need to reserve Option2
I need Name, email, phone
I need date of birth
I need special requests
Do I have all the info?
AGENT
Book Flight
TOOL
bookFlight API
Request more info
No
"Name, email, DOB, Phone, special req"
"Should I confirm booking and email payment link?"
Confirm Flight booking?
SAN to SFO { .... }
SFO to SAN { .... }
PLANNING
"Yes please confirm booking"
I need to confirm booking
I need email payment link
Do I have all the info?
AGENT
Confirm Flight
TOOL
confirmFlight API
AGENT
Email Payment link
TOOL
sendEmail API
"Thanks, booking is confirmed and email has been sent."
Email with Payment email for Booking sent
SAN to SFO { .... }
SFO to SAN { .... }

Tendencias 2025-2030: escalamiento + eficiencia (señales técnicas)

Costo de inferencia en sistemas tipo GPT-3.5: caída >280× (Nov 2022 -> Oct 2024)
A nivel hardware: costos bajando ~30% anual y eficiencia energética subiendo ~40% anual
Modelos open-weight cerrando brechas: en algunos benchmarks pasan de ~8% a ~1.7% en un año
Consequence: despliegue masivo + foco en optimización (cuantización, distilación, batching) y evaluación

Fuente de cifras: AI Index Report 2025 (Stanford HAI).

IA como infraestructura: cómputo, energía y capex (señales macro)

- Demanda eléctrica de data centers: proyección de duplicarse a ~945 TWh en 2030 (Base Case)
- Capex global proyectado para data centers hasta 2030: ~US$6.7T (AI: ~US$5.2T; no-AI: ~US$1.5T)
- Cuellos de botella: capacidad eléctrica, refrigeración, redes, permisos y tiempos de conexión a la red

> Implicación: IA como infraestructura -> el diseño debe incluir energía, coste, seguridad y gobernanza

Fuentes de cifras: IEA (Energy and AI) y McKinsey (The cost of compute).