Arquitectura de Información

aplicada al desarrollo de software

Técnicas de UX y Diseño de Interfaces



Cuatro acciones para transformar información y funcionalidades en una experiencia comprensible.

Carrera: Ingeniería de Sistemas

Equipo docente: Fabricio Defelippe y Luis Berdún

Documento de apoyo académico · Noviembre de 2026

Lectura obligatoria · Material ampliado a partir de la presentación del Módulo 3

Guía de lectura · Del contenido a la estructura

Un sistema puede almacenar datos correctamente, cumplir sus operaciones técnicas y, aun así, hacer que las personas se pierdan. Un menú con nombres internos, un filtro que no corresponde al catálogo o un flujo que obliga a volver al inicio son síntomas de un problema estructural. **La arquitectura de información (IA)** trabaja precisamente sobre esas relaciones: qué existe, cómo se agrupa, cómo se nombra y cómo se encuentra.

El módulo se apoya en un **ejemplo didáctico transversal: una plataforma de reserva de canchas**. Supongamos que permite explorar sedes, consultar disponibilidad, reservar turnos, pagar y cancelar. Las decisiones de IA deberán reflejar esos objetivos sin obligar a las personas a comprender las entidades de la base de datos.

AL FINALIZAR EL MÓDULO

El estudiante debería poder **definir y justificar una estructura de contenidos**, aplicar card sorting y tree testing, construir un sitemap, modelar flujos y estados, y traducir esas decisiones en especificaciones útiles para diseño y desarrollo.

Mapa de contenidos

Bloque	Aprendizaje principal
1 · Fundamentos	Qué es IA, qué resuelve, cuándo utilizarla y cómo se diferencia de UI, sitemap y datos.
2 · Organización	Cinco componentes; clasificación, taxonomías, facetas y vocabularios.
3 · Investigación	Inventario, auditoría, card sorting y tree testing.
4 · Encontrabilidad	Etiquetas, rastro de información, búsqueda, navegación y orientación.
5 · Modelado	Sitemaps, task flows, user flows, CRUD, estados y wireflows.
6 · Integración	Trabajo colaborativo, validación y relación con los próximos prácticos.

Fuente y ampliación: NN/g · Information Architecture: Study Guide

1. Fundamentos de arquitectura de información

1.1 Qué es, qué resuelve y cuándo aplicar IA

La **IA es la disciplina que organiza, estructura, relaciona y etiqueta información y funciones** para facilitar su comprensión y recuperación. No equivale a dibujar un menú: un menú es una representación visible de decisiones estructurales que también afectan búsqueda, permisos, recorridos y mantenimiento.

Se aplica **antes de maquetar** un sitio nuevo, **al rediseñar** una plataforma existente que presenta confusión o contenidos duplicados, y **cuando crece** un producto con más módulos, usuarios o servicios. También es importante al migrar contenido y al integrar distintos sistemas que utilizan nombres incompatibles.

Síntoma del producto	Pregunta de IA que ayuda a resolverlo
Las personas no encuentran “cancelar reserva”.	¿Dónde esperan encontrar esa acción y bajo qué etiqueta?
Una misma cancha aparece en tres categorías distintas.	¿Necesitamos facetas, referencias cruzadas o una regla de clasificación?
El buscador devuelve información irrelevante.	¿Qué metadatos, sinónimos o filtros faltan?
Cada equipo llama distinto al mismo concepto.	¿Cuál es el término preferido y qué variantes deben reconocerse?

Fuente y ampliación: Olga Carreras · Arquitectura de información. Fundamentos

1.2 Tres pilares y tres modelos diferentes

Morville y Rosenfeld proponen comprender la IA a partir de la intersección entre **contexto, contenido y usuarios**. El contexto incluye los objetivos institucionales y restricciones técnicas; el contenido comprende datos, objetos y relaciones; y los usuarios aportan tareas, lenguaje, expectativas y situaciones de uso. Diseñar solo desde una de estas perspectivas puede producir una estructura formalmente correcta, pero poco útil.

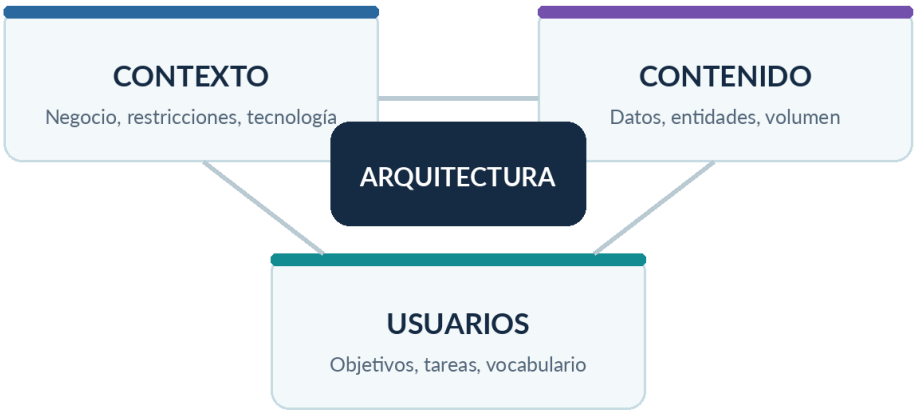

Figura 1. Los tres pilares de IA. Síntesis didáctica de Morville y Rosenfeld.

NN/g distingue además **navegación visible (frontstage)**, **estructura subyacente (backstage)** y **taxonomías y metadatos**. Una persona ve pestañas y filtros; el equipo de diseño define la jerarquía; y el sistema necesita descripciones consistentes para organizar y recuperar entidades. Son capas relacionadas, pero no equivalentes.

Fuente y ampliación: NN/g · IA: 3 Key Models (video y guía de estudio)

1.3 IA, sitemap, navegación y modelo de datos

Concepto	Pregunta que responde	Ejemplo
IA	¿Cómo se organizan, relacionan y nombran las cosas?	Reservas, sedes, canchas, disponibilidad y pagos.
Sitemap	¿Qué pantallas o destinos existen y cómo se vinculan?	Inicio → Explorar → Detalle → Confirmación.
Navegación	¿Cómo llega y se orienta la persona?	Menú global, local, enlaces, breadcrumb.
Modelo de datos	¿Cómo persiste y relaciona la información el backend?	Tablas, claves y reglas de integridad.

La estructura de IA **influye** en las decisiones de ingeniería, pero no determina por sí sola el esquema físico de la base de datos ni la forma exacta de una API. Un filtro por “superficie” puede resolverse de más de una manera técnica. Lo importante es conservar el significado acordado entre el dominio, la interfaz y los datos.

IDEA CENTRAL

Una pantalla puede ser visualmente atractiva y presentar mala IA. La calidad se comprueba observando si las personas **encuentran, comprenden y completan tareas**, no contando elementos decorativos.

2. Anatomía: los cinco componentes de IA

Olga Carreras, a partir de Pérez-Montoro y de Rosenfeld y Morville, describe **cinco componentes**: sistemas de organización, etiquetado, navegación, búsqueda y vocabularios o lenguajes documentales. Los cuatro primeros se observan frecuentemente en la experiencia visible; los vocabularios ofrecen una base semántica transversal.

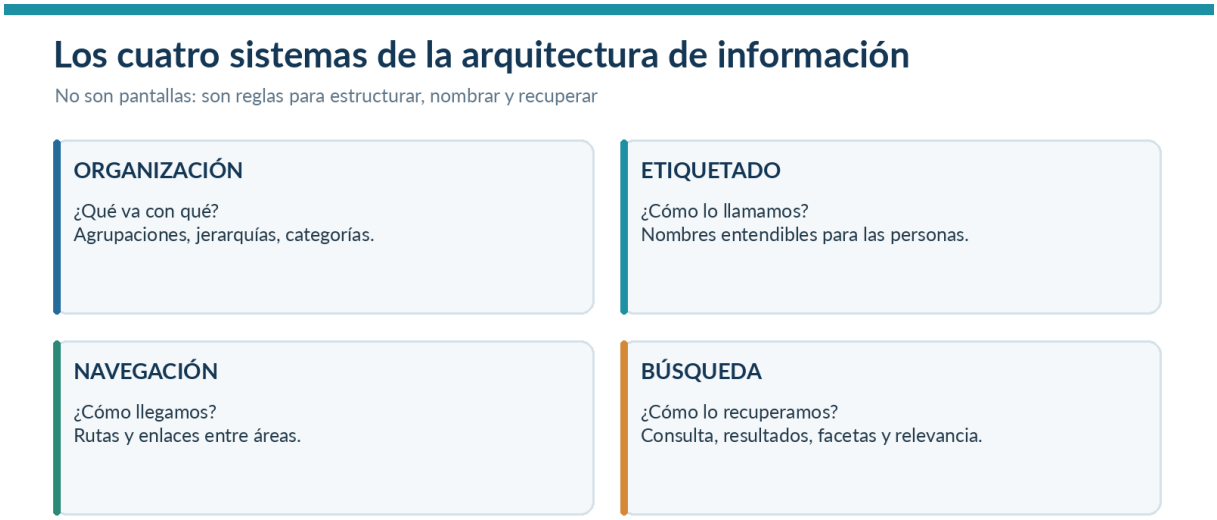

Figura 2. Los cuatro sistemas operativos. Los vocabularios controlados actúan transversalmente sobre ellos.

Componente	Pregunta	En una plataforma de reservas
Organización	¿Qué se agrupa con qué?	Reservas, sedes, ofertas y ayuda.
Etiquetado	¿Cómo se llama cada cosa?	“Mis reservas” y no “ABM Turnos”.
Navegación	¿Cómo nos desplazamos?	Menú, tabs, enlaces, breadcrumbs.
Búsqueda	¿Cómo recuperamos recursos?	Buscar “pádel centro” y filtrar resultados.
Vocabularios	¿Qué términos y sinónimos reconoce el sistema?	“Turno” y “reserva” con significado acordado.

Fuente y ampliación: Olga Carreras · Componentes de IA

3. Organizar, clasificar y jerarquizar

3.1 Esquemas exactos y ambiguos

Un **esquema exacto** utiliza categorías verificables, como orden alfabético, fecha o ubicación. Un **esquema ambiguo** organiza por tema, tarea o audiencia y requiere interpretar qué esperan las personas. Ambos pueden combinarse, siempre que el criterio quede claro. Por ejemplo, una lista por ciudad es exacta; agrupar canchas en “Para jugar con amigos” es interpretativo.

Tipo	Ejemplo	Riesgo o ventaja
Alfabético / cronológico	Clubes A–Z; reservas del mes.	Predecible cuando se conoce el nombre o fecha.
Geográfico	Canchas por barrio.	Útil si la ubicación domina la tarea.
Por tarea	Reservar, modificar, cancelar.	Alinea el sistema con acciones reales.
Por audiencia	Jugadores, clubes, administradores.	Puede crear silos o ambigüedad entre perfiles.
Híbrido	Deporte + ubicación + disponibilidad.	Necesita reglas consistentes para no generar duplicaciones.

Fuente y ampliación: Olga Carreras · Sistemas de organización

3.2 Jerarquías, amplitud, profundidad y polijerarquía

La **amplitud** describe cuántas categorías se muestran por nivel; la **profundidad**, cuántos niveles hay que atravesar. Ni una jerarquía plana ni una profunda son automáticamente mejores: importa la claridad de las etiquetas, el volumen de contenido, la familiaridad de la audiencia y el modo en que se orienta cada paso. NN/g advierte que ambos extremos pueden producir problemas.

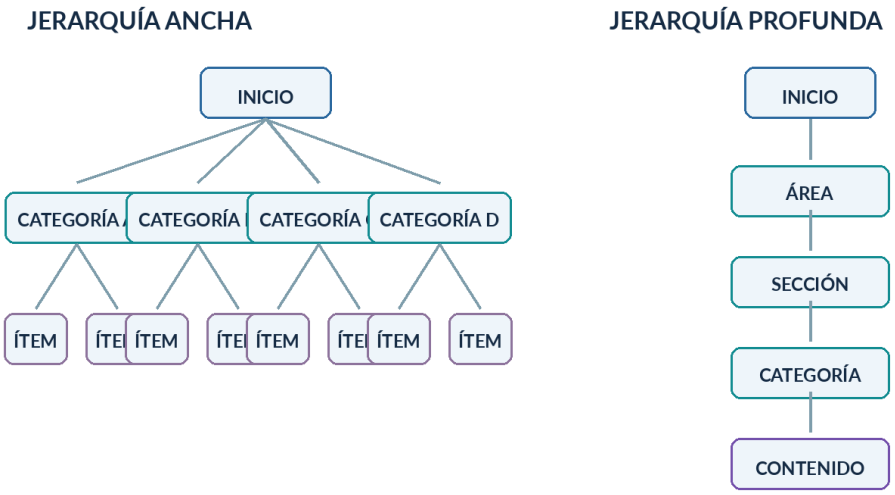

Figura 3. Dos configuraciones posibles de una jerarquía. Esquema didáctico.

La regla de los **tres clics** no debe usarse como criterio universal. Tampoco corresponde trasladar el **7 ± 2** de memoria de corto plazo a un límite fijo de opciones en menús visibles. El éxito depende más de que cada elección tenga sentido que de alcanzar una cifra arbitraria.

Una **polijerarquía** permite llegar al mismo recurso desde varias rutas legítimas: una cancha techada puede aparecer por deporte y por instalaciones cubiertas, sin duplicar el registro subyacente. Requiere gobernar rutas, etiquetas y referencias para evitar inconsistencias.

Fuente y ampliación: NN/g · Flat vs. Deep Website Hierarchies

3.3 Taxonomías, facetas y vocabularios controlados

Una **taxonomía** organiza términos y clases mediante relaciones. Las **facetas** describen dimensiones independientes de un mismo conjunto: deporte, barrio, superficie, precio y horario pueden combinarse sin construir decenas de ramas rígidas. Un **vocabulario controlado** define qué nombre usar, qué sinónimos reconocer y qué relaciones mantener.



Figura 4. Árbol rígido frente a clasificación por facetas de una plataforma de reservas.

01. **Identificar entidades y atributos.** ¿Qué es una cancha? ¿Qué datos describen una sede? ¿Qué caracteriza una reserva?
02. **Separar categorías de atributos.** "Fútbol" puede ser valor de la faceta deporte, no una pantalla obligatoria.
03. **Fijar nombres preferidos y sinónimos.** Por ejemplo "canchas techadas" y "cubiertas".
04. **Resolver excepciones.** Un recurso puede tener más de una disciplina o varios servicios.
05. **Probar la clasificación.** Validar agrupaciones con usuarios y localizar recursos clave en el árbol.

EVITAR

No transformar cada campo de base de datos en una categoría del menú. **Una taxonomía, una faceta y una etiqueta son herramientas diferentes.**

Fuente y ampliación: Olga Carreras · Vocabularios o lenguajes documentales

3.4. Priorizar contenido y comprender distintas necesidades de información

La arquitectura no es una lista alfabética. Para dar prioridad a las categorías conviene considerar frecuencia de uso, impacto de la tarea, contexto de acceso y relación entre recursos. En una plataforma de canchas, «Mis reservas» puede necesitar un acceso permanente aunque alfabéticamente aparezca después de «Canchas»; una política de cancelación, en cambio, debería estar cerca del momento en que se cancela una reserva.

Hay al menos tres situaciones de búsqueda frecuentes: localizar algo conocido («mi reserva del jueves»), explorar alternativas («qué deportes y horarios existen cerca») y completar una operación («cancelar sin penalización»). Cada una impone prioridades distintas a navegación, búsqueda, filtros y enlaces contextuales. No se debe forzar todo el contenido a un único recorrido.

Tampoco toda organización es un árbol: una jerarquía agrupa secciones; una secuencia establece pasos; una red hipertextual conecta recursos relacionados; y una tabla permite comparar opciones con atributos homogéneos. Un checkout es secuencial, un directorio de sedes puede ser jerárquico y una grilla de horarios es tabular.

3.5. Navegación por audiencia y ejemplos de amplitud/profundidad

Agrupar solamente por destinatarios («Estudiantes», «Docentes», «Graduados») es arriesgado cuando las personas pertenecen a más de un grupo o necesitan tareas compartidas. Puede esconder la inscripción y los certificados bajo perfiles confusos. Una arquitectura por tareas suele facilitar «Inscribirme», «Consultar fechas» y «Solicitar certificado»; la separación por audiencias se justifica cuando hay requisitos realmente diferentes.

NN/g ilustra el equilibrio entre amplitud y profundidad con ejemplos históricos de sitios hospitalarios. Tampa General Hospital exponía un conjunto extenso de servicios, lo cual hacía visible la oferta pero imponía un alto esfuerzo de escaneo. UF Health concentraba más información en agrupaciones profundas, reduciendo el volumen simultáneo de opciones pero sumando pasos; también ofrecía accesos directos para tareas frecuentes. Son ejemplos de decisiones con costos diferentes, no de un ganador universal.

Para evaluar una jerarquía, no basta con contar clics: verificar si las etiquetas permiten anticipar el destino, si las personas retroceden, si la categoría esperada es visible y si existen accesos alternativos para recursos importantes. Un menú con siete opciones ambiguas puede ser peor que otro con once opciones bien nombradas.

Fuente y ampliación: NN/g · Flat vs. Deep Website Hierarchies; Audience-Based Navigation; Olga Carreras · Sistemas de organización.

4. Inventario y auditoría de contenidos

4.1 Inventariar: saber qué existe

El **inventario** es una enumeración sistemática de páginas, pantallas, tipos de contenido, campos y rutas. Su objetivo es describir el estado actual sin confundirlo todavía con una evaluación de calidad. Es especialmente útil al rediseñar, migrar o integrar sistemas.

01. **Definir alcance.** Sitio completo, módulo de reservas o grupo concreto de pantallas.
02. **Listar unidades.** URL o ruta, título, tipo, propietario, audiencia, fecha y dependencias.
03. **Registrar atributos estructurales.** Estado, permisos, etiquetas, metadatos, ubicación actual.
04. **Detectar duplicaciones.** Variantes de nombres o páginas que atienden una misma necesidad.
05. **Consolidar el inventario.** Asignar ID y responsable por cada recurso.

Inventario ≠ auditoría de contenido

Ejemplo de registro para el rediseño de una plataforma de reservas

ID	Contenido o pantalla	Responsable	Estado	Decisión
01	Buscar canchas	Producto	Vigente	MANTENER
02	FAQ cancelaciones	Soporte	Desactualizado	ACTUALIZAR
03	Promociones 2022	Marketing	Obsoleto	RETIRAR
04	Reglas de reembolso	Legal	Incompleto	COMPLETAR

Inventario: qué existe y dónde. Auditoría: qué calidad tiene y qué hacemos con cada pieza.

Figura 5. Fragmento de un inventario de contenidos: registro antes de tomar decisiones.

4.2 Auditar: decidir qué conservar y mejorar

La **auditoría** analiza si el contenido es vigente, útil, correcto, completo, encontrable y consistente. Conviene separar el diagnóstico de la acción: "desactualizado" es una evidencia; "retirar" o "reescribir" es la decisión posterior. Una matriz puede utilizar las categorías **mantener, actualizar, consolidar, crear y retirar**.

ID	Hallazgo del contenido	Decisión / evidencia requerida
C-04	Dos páginas distintas explican la política de cancelación.	Consolidar y verificar la regla con el área operativa.
C-11	El usuario busca horarios, pero la página se llama "Servicios".	Revisar etiqueta y validar en un tree test.
C-15	La ayuda sobre pagos está incompleta.	Actualizar responsable, contenido y fecha.

ENTREGABLE PRÁCTICO

Inventario → auditoría → propuesta de estructura. No diseñar un sitemap solo desde la intuición del equipo.

4.3. Auditoría del sistema de búsqueda: precisión y exhaustividad

Además de auditar páginas, conviene probar las consultas reales. La precisión disminuye cuando el buscador devuelve demasiados resultados que no responden a la intención: buscar «canchas de fútbol» y obtener también salones de fiestas. La exhaustividad o recall disminuye cuando omite resultados relevantes: buscar «F5» y no encontrar una cancha indexada solamente como «fútbol 5». Ambas fallas pueden coexistir.

Procedimiento: seleccionar consultas conocidas, sinónimos y faltas frecuentes; definir qué resultados deberían aparecer; ejecutar cada búsqueda en dispositivos relevantes; registrar el conjunto recuperado, los resultados irrelevantes y las omisiones; analizar metadatos, vocabulario, ranking, filtros y mensajes de «sin resultados»; priorizar correcciones y repetir las pruebas. No se deben calcular porcentajes de precisión o recall si no se conoce el universo de resultados relevantes.

Ejemplo de registro: «La consulta "pádel" devuelve cinco canchas de tenis y omite una sede de pádel. Evidencia: captura, hora, consulta exacta y resultados esperados. Hipótesis: el índice no reconoce el sinónimo ni la categoría. Acción propuesta: normalizar el vocabulario y revisar el ranking».

Fuente y ampliación: Olga Carreras · Sistemas de búsqueda y vocabularios controlados; NN/g · Information Architecture: Study Guide.

5. Card sorting: descubrir cómo agrupan las personas

En **card sorting** se entregan tarjetas que representan páginas, recursos o tareas y se pide a participantes que las agrupen de manera significativa. Es una técnica de investigación **generativa** para comprender modelos mentales; no define por sí sola el menú definitivo.

5.1 Abierto, cerrado y mixto

Modalidad	Qué hace el participante	Uso recomendado
Abierto	Agrupa libremente y pone nombre a sus conjuntos.	Explorar agrupaciones y vocabulario.
Cerrado	Ubica tarjetas en categorías previamente definidas.	Comprobar una clasificación candidata.
Mixto / híbrido	Usa categorías propuestas y puede crear nuevas.	Refinar una estructura inicial.



Figura 6. Card sorting abierto y cerrado: procedimientos distintos para preguntas distintas.

5.2 Procedimiento paso a paso

01. **Formular la pregunta de investigación.** Por ejemplo: "¿cómo agrupan las personas la información de reservas, pagos y ayuda?".
02. **Elegir tarjetas representativas.** Usar contenido real y descripciones inequívocas; evitar pistas que ya revelen la categoría.
03. **Reclutar participantes relevantes.** Para un estudio formal se necesita una muestra planificada; pocos compañeros sirven solo como práctica exploratoria.
04. **Ejecutar individualmente.** Pedir que agrupen, nombren y expliquen dudas sin sugerir respuestas.
05. **Registrar discrepancias.** Las tarjetas que cambian de grupo suelen revelar ambigüedades valiosas.
06. **Analizar patrones.** Revisar matriz de similitud, agrupaciones frecuentes y nombres propuestos.
07. **Proponer una estructura candidata.** Construir categorías y comprobarlas después con tree testing.

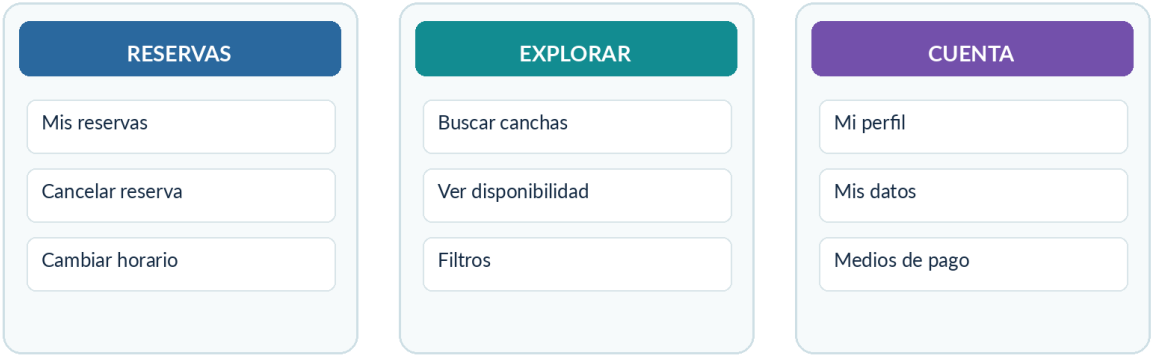

Figura 7. Ejemplo didáctico de posibles grupos surgidos de un card sorting.

Fuente y ampliación: NN/g · Card Sorting: Uncover Users' Mental Models

5.3 Interpretación: matriz de similitud y dendrograma

Una **matriz de similitud** muestra con qué frecuencia dos tarjetas fueron colocadas juntas; un **dendrograma** ofrece una representación de agrupaciones calculadas. Ambos requieren interpretación: no es correcto convertir mecánicamente cada clúster en una categoría del menú.

Si “Cancelar reserva” aparece en «Mis reservas», «Ayuda» y «Políticas», no necesariamente hay un participante equivocado. Puede indicar que la tarea admite varios puntos de acceso, que su etiqueta es ambigua o que el alcance de las categorías no está bien delimitado. La decisión se fundamenta posteriormente con pruebas de encontrabilidad.

ERROR FRECUENTE

Si todas las tarjetas contienen “reserva”, un grupo resultante podría reflejar coincidencia de palabras y no un modelo mental útil. Redactar tarjetas de forma neutral y proporcional al contenido.

Fuente y ampliación: NN/g · Card Sorting: Pushing Users Beyond Terminology Matches

5.4. Card sorting: tamaños, sesgos y lectura crítica de los resultados

En un estudio profesional, la cantidad de tarjetas y participantes depende de la variedad del contenido y del nivel de certeza buscado. Como orientación metodológica, los ejemplos de NN/g suelen trabajar con conjuntos de decenas de tarjetas y muestras mayores que una práctica de clase. Un ejercicio con pocos compañeros permite aprender la técnica, pero no justificar decisiones universales sobre toda la audiencia.

Un riesgo común es que las personas agrupen elementos que comparten una palabra sin tener propósitos relacionados: «Employee Directory», «Employee Benefits» y «Employee Training» pueden parecer similares solo porque repiten «Employee». Antes de decidir categorías, hay que leer qué significa cada recurso y escuchar por qué se agrupó. La consistencia verbal no prueba cercanía funcional.

La matriz de similitud muestra cuántas personas agruparon juntas dos tarjetas. Si un par se agrupó por el 86 % de los participantes, existe una señal fuerte de afinidad en esa muestra; no es una obligación de diseñar una categoría. Un dendrograma organiza visualmente proximidades estadísticas, pero no decide el menú. Las tarjetas que cambian de grupo, la categoría «Otros» y las explicaciones cualitativas merecen revisión particular.

Interpretación aplicada: ante una tarjeta «Cancelar reserva» repartida entre «Mis reservas», «Ayuda» y «Políticas», plantear al menos dos estructuras posibles, comprobar su etiquetado y usar tree testing. El card sorting ayuda a descubrir modelos mentales; no demuestra que una ruta concreta vaya a encontrarse con éxito.

Fuente y ampliación: NN/g · Card Sorting: Pushing Users Beyond Terminology Matches; How to Interpret Dendrograms; Card Sorting: Uncover Users’ Mental Models.

6. Tree testing: validar la estructura sin interfaz

El **tree testing** presenta una jerarquía de categorías en forma de texto, sin colores, íconos ni diseño visual. Se pide localizar recursos a partir de tareas realistas. Sirve para verificar si los nombres y ubicaciones del árbol permiten encontrar contenido **antes de invertir tiempo en maquetas de alta fidelidad**.

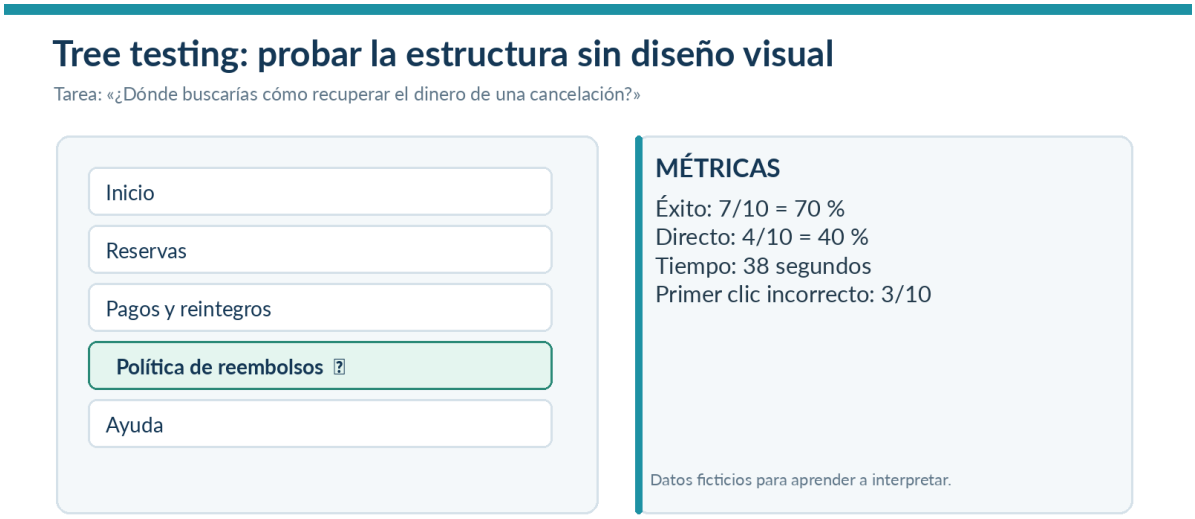

Figura 8. Ejemplo de tree testing: tarea, árbol simplificado y métricas ficticias de ilustración.

6.1 Procedimiento paso a paso

01. **Construir el árbol.** Incluir las categorías hasta el nivel donde se espera encontrar el recurso.
02. **Elegir recursos críticos.** Incluir tareas frecuentes, rutas ambiguas y nuevas categorías.
03. **Redactar tareas naturales.** “Necesitás cambiar la fecha de una reserva ya realizada”; no decir “Entrá a Mis reservas”.
04. **Definir destinos válidos.** Documentar dónde se espera que las personas lleguen.
05. **Ejecutar el test.** Registrar primer clic, retrocesos, destino y tiempo.
06. **Interpretar resultados.** Medir éxito, rutas directas, elecciones erróneas y causas posibles.
07. **Iterar.** Revisar etiquetas o estructura y repetir el estudio sobre los puntos conflictivos.

6.2 Qué significan las métricas

Medida	Interpretación
Éxito	Proporción que llega a una ubicación aceptada como correcta.
Éxito directo	Llega sin desviarse hacia ramas equivocadas.
Primer clic	Primera hipótesis sobre dónde vive el contenido.
Retrocesos	Cantidad o patrón de exploraciones incorrectas.
Tiempo	Esfuerzo aproximado, interpretado junto con el éxito y el recorrido.

EJEMPLO DE LECTURA

Si 8 de 10 participantes llegan, pero 6 entran primero en una categoría incorrecta, el árbol **no está resuelto**: el éxito final puede esconder dudas o recorridos costosos. Estos números son ficticios y se utilizan solo para ilustrar.

Fuente y ampliación: NN/g · Tree Testing: Evaluate Menu Labels and Categories

6.3. Tree testing: cómo leer trayectorias y diagnosticar errores

El resultado no debería resumirse únicamente como «80 % de éxito». Hay que distinguir el acierto directo de quien llega por una ruta breve y razonable, del acierto indirecto de quien se equivoca, retrocede y finalmente encuentra el recurso. Ambos terminan bien, pero el segundo indica mayor esfuerzo y posible ambigüedad en las primeras categorías.

Al revisar una tarea, observar primera elección, retrocesos, destinos erróneos, abandono y tiempo. Un primer clic repetidamente equivocado puede revelar una etiqueta engañosa; un éxito final alto acompañado de múltiples retrocesos puede esconder una arquitectura que obliga a explorar a ciegas.

Antes de cambiar el árbol conviene distinguir si el problema está en la estructura o en su representación visual. Si el árbol textual de un tree test funciona pero el sitio real falla, revisar señales de orientación, jerarquía visual, componentes ocultos y enlaces. Si ambos fallan en el mismo nodo, revisar categorías y etiquetas; es un indicio de un problema de IA, no una prueba causal definitiva.

Ejemplo: «Necesitás cambiar la fecha de una reserva». Si el participante entra primero a «Ayuda» en vez de «Mis reservas», registrar la ruta. Una recomendación posible es habilitar un enlace contextual desde la reserva y comprobar después si resuelve la tarea, sin suponer que un único cambio será suficiente.

Fuente y ampliación: NN/g · Tree Testing: Evaluate Menu Labels and Categories; IA vs. Navigation Problems.

7. Etiquetado, búsqueda y encontrabilidad

7.1 Las palabras que usa la gente

Una etiqueta es una promesa sobre el contenido que se encontrará después de actuar. Debe utilizar palabras del dominio de las personas, no abreviaturas que solo comprenden los desarrolladores. Una sección titulada **“ABM Reservas”** puede resultar natural para quien la programó, mientras que **“Mis reservas”** comunica mejor su propósito.



Figura 9. Ejemplos de jerga interna y traducción a lenguaje de tareas.

01. Reunir nombres existentes en entrevistas, búsqueda interna, soporte y contenidos.

02. Comparar términos para una misma tarea y detectar polisemia o duplicados.

03. Elegir una etiqueta concreta y coherente con la audiencia.

04. Probar su interpretación en contexto mediante tareas y primeras selecciones.

05. Registrar términos preferidos, sinónimos y reglas de uso en un glosario del producto.

7.2 Findability, discoverability e information scent

Findability (encontrabilidad) refiere a poder localizar un recurso buscado; **discoverability (descubribilidad)**, a notar una capacidad que quizá ni se sabía que existía. Son problemas distintos. Pedir a alguien “encontrá el

chat" comprueba principalmente encontrabilidad; observar si descubre espontáneamente un chat de ayuda evalúa otra condición.

El **information scent** o rastro de información son las pistas que permiten estimar si un enlace conducirá a algo relevante: etiqueta, contexto, descripción e historia previa. "Soluciones" es vago; "Ver disponibilidad de canchas" anticipa una consecuencia concreta. Su calidad depende de la tarea y del usuario.



Figura 10. Rastro débil frente a enlaces de propósito explícito.

Fuente y ampliación: NN/g · Information Scent: How Users Decide Where to Go Next

7.3 Buscar no reemplaza navegar

La **búsqueda** es útil cuando se dispone de términos concretos, mientras que la **navegación** permite explorar el alcance del sistema y reconocer categorías posibles. Las personas alternan entre ambas. La teoría del forrajeo de información y el modelo *berrypicking* explican cómo reformulan una necesidad a medida que descubren recursos relacionados.

Un buscador requiere decisiones estructurales: qué se indexa, qué sinónimos se reconocen, cómo se ordenan resultados, qué sucede sin coincidencias y qué filtros existen. Una búsqueda de "paddle" debería contemplar la variante "pádel" si es pertinente. También conviene medir consultas sin resultados y reformulaciones frecuentes.

Fuente y ampliación: Olga Carreras · Sistemas de búsqueda

7.4. Enlaces con buen rastro: las cuatro S y sus límites

El etiquetado puede revisarse con las cuatro S presentadas en los slides a partir de NN/g: Specific (indica el destino concreto), Sincere (promete honestamente lo que ocurrirá al hacer clic), Substantial (es comprensible incluso leído fuera del párrafo que lo rodea) y Succinct (usa pocas palabras, pero las necesarias). No basta con tener pocos caracteres: el enlace también debe ser informativo y fiel a su destino. Estos criterios orientan el análisis, pero no reemplazan probar etiquetas con usuarios.

Comparar «Conocer más» con «Ver condiciones de cancelación»: el segundo anticipa qué se encontrará y en qué contexto será útil. En GOV.UK, muchos enlaces se redactan a partir de tareas reales de los ciudadanos y no del nombre de un departamento. Al evaluar, pedir que una persona diga qué espera encontrar antes de hacer clic.

Conviene realizar dos pruebas diferentes. Para encontrabilidad, pedir una tarea con destino conocido sin revelar la etiqueta («Necesitás modificar un turno para mañana»). Para descubribilidad, observar si una persona advierte espontáneamente una funcionalidad nueva durante una tarea natural; nombrarla de antemano invalida esa segunda prueba.

Fuente y ampliación: NN/g · Information Scent; Link Design: 4 Ss; GOV.UK en casos de IA.

8. Navegación y orientación

8.1 Sistemas global, local, contextual y complementario

La navegación **global** orienta entre grandes áreas del producto; la **local** recorre las opciones de una sección; la **contextual** conecta contenidos relacionados; y la **complementaria** incorpora mapas del sitio, índices, guías o ayudas especiales. Deben coexistir sin competir por la atención ni cambiar arbitrariamente de ubicación.



Figura 11. Navegación global, local y contextual en una misma página.

8.2 “You are here”: cómo saber dónde estamos

En productos extensos, orientarse es parte de poder navegar. NN/g recomienda ofrecer pistas consistentes sobre la ubicación actual: **título visible, opción activa, breadcrumbs, URL descriptiva y pasos de proceso**, según el caso. No alcanza con poner enlaces; una página debe transmitir dónde está la persona incluso si llega directamente desde un buscador.



Figura 12. Varias pistas de orientación coexistiendo en una misma pantalla. Ejemplo didáctico.

01. Abrir una página interior sin entrar por la portada del sitio.

02. Preguntar: ¿dónde estás?, ¿en qué sección?, ¿qué otras opciones pertenecen a esta área?

03. Registrar qué pista utiliza: encabezado, menú activo, breadcrumb o contexto.

04. Repetir con pantallas de distinta profundidad y desde el móvil.

05. Corregir señales ausentes o inconsistentes antes de añadir más niveles.

Fuente y ampliación: NN/g · Navigation: You Are Here

8.3 Cuántas opciones mostrar y cuándo ocultar menús

Ocultar opciones no elimina complejidad: puede dificultar descubrir qué ofrece el sistema. Un menú hamburguesa es un patrón reconocible, pero su conveniencia depende del dispositivo, la frecuencia de uso y el costo de ocultar categorías. En escritorio, la navegación persistente suele hacer más visibles los destinos principales; en móviles debe priorizarse qué exponer y qué agrupar.

Los **mega menús** permiten mostrar familias de contenido en un solo despliegue; los **menús verticales** ayudan en estructuras amplias; los **acordeones** pueden organizar categorías locales. Ninguno sustituye una buena taxonomía: solo presenta una estructura existente.

Fuente y ampliación: NN/g · Menu Design: Checklist of 15 UX Guidelines

8.4. Wayfinding: referencias, rutas y pruebas de orientación

La orientación digital puede pensarse por analogía con el wayfinding de espacios físicos: puntos de referencia constantes (logo y nombre del módulo), distritos reconocibles (Explorar, Reservar, Administrar), nodos de decisión (menús y pantallas de categorías), rutas (enlaces y breadcrumbs) y límites entre áreas. Son analogías didácticas, no coordenadas físicas del sistema.

NN/g presenta ejemplos históricos de BBC News y MIT OpenCourseWare donde títulos, sección activa, navegación lateral y breadcrumbs ofrecen señales redundantes para situar a la persona. Una página interior debe poder entenderse incluso cuando alguien llega por búsqueda, notificación o enlace externo, sin pasar antes por la portada.

Al analizar una pantalla, diferenciar cuatro ubicaciones habituales: navegación global, local, utilitaria (cuenta, idioma, soporte) y footer (enlaces secundarios y legales). No son cuatro jerarquías obligatorias: son roles diferentes que pueden coexistir. La navegación complementaria agrega mapas, índices y ayudas, sin reemplazar una taxonomía comprensible.

8.5. Elegir menús visibles, verticales, mega menús y subnavegación móvil

Ocultar un menú puede dar protagonismo al contenido visual, pero también reducir el reconocimiento de destinos disponibles. NN/g estudió diferencias entre menús visibles, ocultos y mixtos: los resultados históricos muestran que la visibilidad modifica el uso de la navegación; no deben interpretarse como una tasa universal para cualquier producto actual. Conviene contrastar el diseño con el contexto y el dispositivo.

En productos amplios, un menú lateral permite muchas opciones locales persistentes, a costa de consumir espacio horizontal. Un mega menú expone varias familias y subcategorías simultáneamente, pero puede convertirse en una pantalla saturada si se añaden demasiados controles. En móvil, acordeones, submenús secuenciales y páginas de categoría permiten repartir niveles; cada uno necesita un retorno y una indicación de profundidad claros.

Los indicadores de expansión importan. Un caret orientado hacia abajo suele sugerir que una sección se puede desplegar, mientras que una flecha hacia la derecha puede interpretarse como navegar a otra página. El signo más exige coherencia al transformarse en menos. Los casos históricos de JennAir, Bolé Road, Ferrari y Braun ilustran cómo el icono elegido altera la expectativa; el problema se evalúa observando qué anticipa la persona antes del clic.

Fuente y ampliación: NN/g · Navigation: You Are Here; Flat vs. Deep Website Hierarchies; Hidden Navigation and Hamburger Menus; Accordion Icons; Vertical Navigation; Menu Design Checklist.

9. Representar la arquitectura: sitemaps y recorridos

9.1 Mapa del sitio: destinos y relaciones

Un **sitemap** representa páginas, secciones o destinos y sus relaciones jerárquicas. Es útil para alinear diseño, contenido y desarrollo. No es un diagrama de todas las interacciones ni un esquema de tablas de base de datos. Conviene incluir identificadores consistentes, rutas, niveles, permisos y destinos alternativos cuando correspondan.



Figura 13. Mapa de sitio para el caso transversal de reservas de canchas.

01. Listar destinos identificados en el inventario y en los objetivos de las personas.

02. Definir agrupaciones y relaciones principales.

03. Representar jerarquía, enlaces transversales y accesos alternativos.

04. Marcar páginas públicas, privadas o restringidas.

05. Revisar con stakeholders y validar rutas críticas mediante tree testing.

9.2 Task flow frente a user flow

El **task flow** describe el camino de una tarea en condiciones predefinidas, normalmente una secuencia principal. El **user flow** incorpora decisiones y variantes, como ingresar sin sesión, fallar un pago o volver a elegir horario. Los dos son útiles, pero responden preguntas distintas.

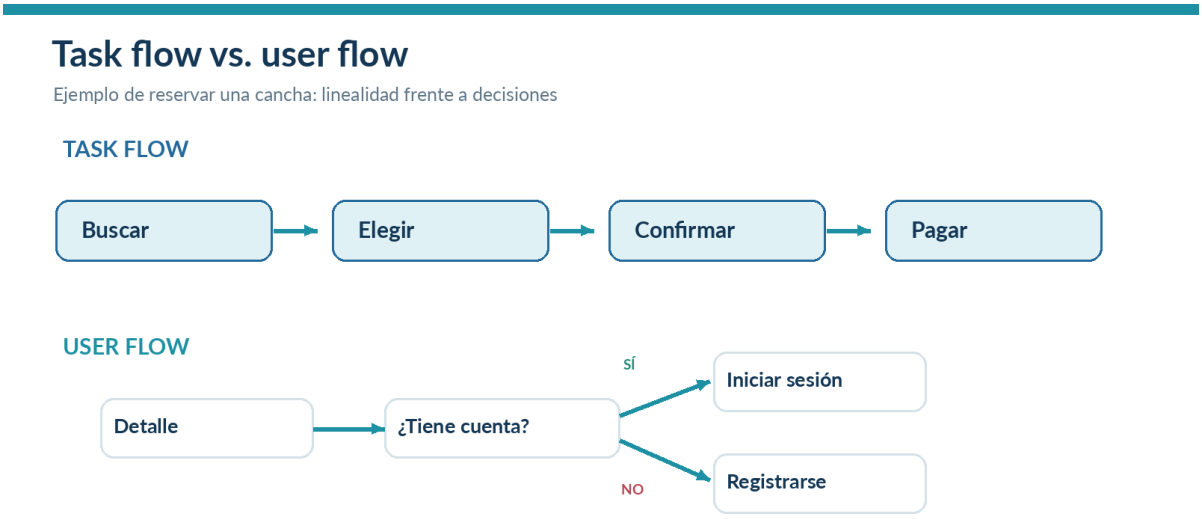

Figura 14. Task flow lineal y user flow con una decisión alternativa.

01. Escribir la tarea, el objetivo y el estado inicial.

02. Identificar las acciones, decisiones, validaciones y resultados.

03. Dibujar el caso principal con pocas convenciones.

04. Agregar variantes que tengan consecuencias relevantes.

05. Comprobar que cada rama termina en resultado o recuperación definidos.

9.3 Modelado de recorridos de usuario

No todo recorrido comienza en Inicio. Una persona puede llegar desde una notificación, una búsqueda externa o un vínculo compartido. El modelado de recorridos debe considerar entradas alternativas, permisos, interrupciones, vuelta atrás y recuperación de contexto. El nivel de detalle debe permitir discutir decisiones, no producir diagramas ilegibles.

> **PREGUNTA PARA EL EQUIPO**
> Si una persona inicia una reserva, pierde conectividad y vuelve diez minutos después, ¿debe comenzar desde cero? Esta decisión conecta **IA, interacción, estados y reglas del producto**.

9.4. Diagramas legibles: cómo separar estructura y comportamiento

Un diagrama deja de ser útil cuando nadie puede revisarlo. Para evitarlo, elegir una sola pregunta por representación: el sitemap muestra destinos; un task flow, la secuencia típica de una tarea; un user flow, condiciones y alternativas. Si aparecen decenas de nodos y flechas cruzadas, conviene dividir en subprocesos y proporcionar referencias entre ellos.

Ejemplo: en lugar de un único lienzo con cincuenta pantallas, modelar por separado «Buscar cancha», «Crear reserva» y «Cancelar reserva». Señalar eventos de entrada y salida, estados compartidos y dependencias. Conservar nombres coherentes entre diagramas, pantallas y documentación funcional.

10. Formularios, CRUD y estados de interfaz

10.1 Diseñar tareas de alta, baja y modificación

Un CRUD no es una lista de cuatro botones. Cada operación posee datos obligatorios, permisos, confirmaciones, restricciones, estados y consecuencias. “Eliminar reserva” puede requerir comprobar un plazo o confirmar un reembolso. La interfaz debe utilizar el lenguaje y las reglas de la tarea, no mostrar sin mediación operaciones técnicas.



Figura 15. Operación CRUD como secuencia de lectura, modificación, validación y confirmación.

Operación	Decisiones y excepciones
Crear	¿Qué datos requiere? ¿Qué sucede si el horario ya fue ocupado?
Consultar	¿Qué información debe mostrar cada perfil? ¿Cómo se filtra?
Actualizar	¿Qué campos pueden cambiarse? ¿Qué conserva el sistema?
Eliminar / cancelar	¿Se elimina realmente, se archiva o cambia de estado? ¿Puede revertirse?

10.2 Formularios que expresan la estructura de la tarea

Una buena arquitectura también se percibe en formularios: agrupar campos por significado, usar etiquetas consistentes, revelar información cuando sea necesaria y conservar datos ante errores. Deben contemplarse validación, dependencias, ayudas contextuales y mensajes precisos. Un proceso de alta puede ser de una sola pantalla o de varias etapas según las decisiones reales, no por preferencia estética.

10.3 Estados: carga, vacío, error, éxito y retorno

Diseñar únicamente el caso ideal deja incompleta la experiencia. Una lista de reservas puede estar cargando, vacía, mostrar datos, fallar o volver de una acción. Cada estado necesita contenido, acciones disponibles y mecanismos de recuperación. Los estados son parte de la especificación funcional y de QA.

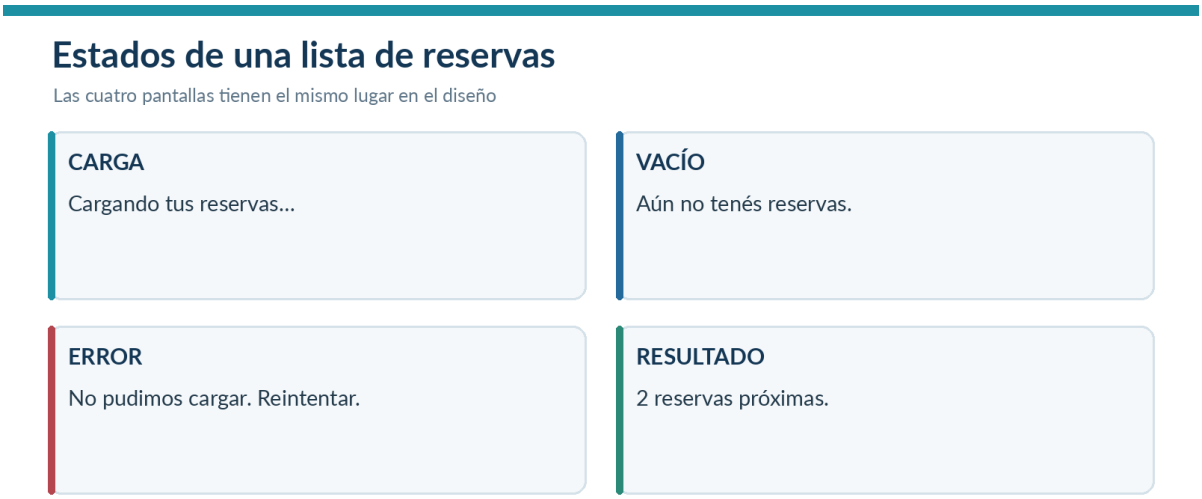

Figura 16. Cuatro estados esenciales de una lista de reservas.

Estado	Qué comunicar	Acción útil
Carga	Se está recuperando la información.	Mantener contexto y evitar repetición accidental.
Vacío	Aún no existen reservas.	Explorar canchas o crear una reserva.
Error	No se pudo recuperar o guardar.	Reintentar y explicar si se conservaron cambios.
Resultado / retorno	La operación terminó y este es el nuevo estado.	Ver detalle, descargar comprobante o deshacer si corresponde.

10.4. Errores de formulario, estados y recuperación de una operación

Una buena validación de formulario comunica el problema junto al campo afectado, explica cómo corregirlo y conserva lo que ya se completó. En lugar de «Error 400», conviene «Seleccioná una fecha futura» o «Revisá el número de contacto». Si hay varios errores, se puede sumar un resumen superior sin sustituir el mensaje local.

En un error transaccional el sistema también debe explicar qué ocurrió con la operación y qué permanece vigente. Si un pago falla, indicar si el horario sigue reservado temporalmente, si hubo un cobro, cuánto tiempo queda y qué acciones son posibles («Reintentar», «Cambiar medio de pago», «Volver a reserva»). No inventar promesas de reserva o reembolso que el backend no pueda garantizar.

Una matriz de estados debería registrar para cada pantalla: evento desencadenante, mensaje visible, acción posible, cambios en datos, condición de salida y criterio para QA. En una lista de reservas: carga («Buscando canchas…»), vacío («Sin resultados: cambiá filtros»), error («No pudimos cargar: Reintentar»), contenido y éxito («Reserva confirmada: Ver comprobante»).

Fuente y ampliación: NN/g · Error-Message Guidelines; GOV.UK Design System · Validation Errors (referencias complementarias de diseño de formularios).

11. Wireflows y especificación para desarrollo

11.1 Unir pantalla, decisión y comportamiento

Un **wireflow** combina wireframes de las pantallas con transiciones y decisiones. A diferencia de un sitemap, describe qué sucede cuando la persona actúa. Es especialmente útil cuando los formularios, estados o permisos generan rutas que deben documentarse antes del desarrollo.



Figura 17. Wireflow del ejemplo transversal: de búsqueda a detalle y reserva.

01. Elegir una tarea crítica y sus estados iniciales.

02. Crear bocetos simples de cada pantalla relevante.

03. Dibujar acciones y condiciones entre pantallas.

04. Agregar errores, permisos, vacíos y rutas de retorno.

05. Vincular cada transición con reglas funcionales o historias de usuario.

06. Revisar con producto, diseño, desarrollo y QA.

11.2 Traducir diagramas en requisitos verificables

Una decisión visual necesita un resultado comprobable. Por ejemplo: “Cuando el usuario intenta reservar un horario ocupado, el sistema no crea la reserva, mantiene la selección de sede y muestra alternativas disponibles”. Esto se puede probar, mientras que “mostrar un error claro” es insuficiente como criterio de aceptación.

Diseño	Regla observable para desarrollo / QA
Filtro por superficie	El listado refleja únicamente canchas con el atributo seleccionado.
Breadcrumb	Cada enlace superior lleva al destino correspondiente sin perder permisos.
Cambio de horario	Si falla la operación, el sistema conserva la reserva original.
Búsqueda sin resultados	Muestra los términos utilizados y alternativas útiles.

12. Trabajo colaborativo y aplicación integradora

12.1 Herramientas: elegir según la pregunta

Necesidad	Herramientas posibles	Entregable

Inventario y auditoría	Planilla compartida, CMS, revisión de URLs.	Catálogo con estado y responsables.
Card sorting	Tarjetas físicas, FigJam, Optimal Workshop.	Agrupaciones, etiquetas y patrones.
Tree testing	Treejack, prototipo textual o herramienta propia.	Rutas, éxito y errores de clasificación.
Sitemap y flujos	Figma/FigJam, Miro, diagrams.net.	Árbol, rutas y decisiones.
Implementación	Tablero de producto, tickets y documentación técnica.	Requisitos, criterios y estados.

La herramienta no valida una arquitectura. Un diagrama bonito no sustituye una prueba de encontrabilidad, y una matriz de card sorting no reemplaza a la interpretación del investigador. Conviene guardar versiones, decisiones, fuentes y desacuerdos.

12.2 Caso transversal: del TP1 al diseño

Retomando el problema de reserva de canchas, el camino razonable es convertir la investigación previa y el benchmark en una estructura candidata. Primero se define el contenido; luego se organiza con evidencia; después se prueba la jerarquía; finalmente se modela la interacción y sus estados.

Etapa	Acción	Evidencia o producto
Descubrir	Revisar entrevistas, necesidades y referentes.	Hallazgos del TP1 y benchmark del TP2.
Organizar	Inventario, vocabulario, taxonomía y facetas.	Propuesta de IA documentada.
Validar	Card sorting y tree testing para rutas clave.	Problemas detectados y decisiones revisadas.
Modelar	Sitemap, user flows, formularios y wireflows.	Especificación para UI, desarrollo y QA.

12.3 Lista de verificación antes de diseñar pantallas

- ¿Sabemos **qué contenidos y funciones** debe ofrecer el sistema y quién los utiliza?
- ¿Cada categoría tiene un **criterio de pertenencia** comprensible?
- ¿Las etiquetas coinciden con el lenguaje de las personas?
- ¿Existen vías razonables para **buscar, explorar y volver**?
- ¿Se probó al menos una ruta crítica sin depender del diseño visual?
- ¿Están modelados los casos alternativos, errores, permisos y retorno?
- ¿Es posible rastrear cada decisión hasta un hallazgo, una necesidad o una regla funcional?

SÍNTESIS DE LA UNIDAD

Diseñar arquitectura de información es **hacer explícitas las relaciones del producto**, justificarlas con evidencia y traducirlas en una experiencia que las personas puedan recorrer. La pantalla es una consecuencia de esa estructura, no su punto de partida.

12.4. Ayuda contextual y colaboración para mantener la IA

La arquitectura también incluye recursos de ayuda. Una política de cancelación debería ser accesible desde «Mis reservas» sin obligar a recorrer la home; una búsqueda de ayuda puede reconocer «turno», «cita» y «reserva» como términos relacionados. La ayuda situada reduce el cambio de contexto, pero no justifica ocultar problemas estructurales detrás de tutoriales.

En colaboración, la trazabilidad es fundamental: investigación identifica problemas; diseño propone categorías y etiquetas; producto prioriza; desarrollo valida restricciones y datos; QA comprueba rutas, permisos, estados y transiciones. Cada cambio importante debería registrar quién lo decidió, evidencia utilizada, fecha y una forma de revisarlo en el siguiente ciclo.

Los videos de NN/g vinculados desde los slides complementan la lectura: modelos de IA, dendrogramas, wayfinding, menús y modalidades de card sorting. No se debe atribuirles contenido oral específico sin disponer de una transcripción verificable.

Fuente y ampliación: NN/g · IA Study Guide; Olga Carreras · Arquitectura de Información: Fundamentos.

13. Glosario y preguntas de repaso

Concepto	Definición operativa
Arquitectura de información	Sistema de decisiones sobre organización, relaciones, etiquetas y acceso al contenido.
Taxonomía	Clasificación con relaciones definidas entre categorías o conceptos.
Faceta	Dimensión independiente para describir o filtrar un conjunto de objetos.
Polijerarquía	Un mismo contenido puede encontrarse por distintas rutas válidas.
Card sorting	Estudio de cómo las personas agrupan y nombran recursos.
Tree testing	Evaluación de una jerarquía textual a través de tareas de búsqueda.
Findability	Capacidad de localizar contenido o función que se busca.
Discoverability	Capacidad de advertir funciones o recursos que no se conocían.
Information scent	Pistas que permiten anticipar el valor de seguir un enlace o categoría.
Sitemap	Modelo de destinos y relaciones estructurales.
User flow	Representación de rutas con decisiones y resultados alternativos.
Wireflow	Pantallas esquemáticas conectadas por acciones y transiciones.

Preguntas para comprobar comprensión

01. ¿Qué problema puede solucionar la IA que no se resuelve cambiando colores o tipografía?

02. ¿En qué se diferencia un vocabulario controlado de un menú de navegación?

03. ¿Por qué no es correcto imponer la regla de tres clics o un número fijo de opciones?

04. ¿Qué resultados ofrece card sorting y qué no puede demostrar?

05. ¿Qué evidencia buscarías en un tree test con alto éxito, pero muchos retrocesos?

06. ¿Qué pistas debe mostrar una pantalla para comunicar “estás aquí”?

07. ¿Cómo se distingue un sitemap de un user flow?

08. ¿Qué reglas y estados habría que documentar para cancelar una reserva sin ambigüedades?

13.1. Cómo utilizar este apunte con las diapositivas

El material sigue el orden de la presentación, pero no pretende repetir el texto de cada lámina. Las capturas históricas, ejercicios interactivos, figuras de comparación y videos se desarrollan aquí mediante la técnica o el problema que ilustran. Para asegurar que ningún slide quede sin correspondencia, el archivo «Auditoría de cobertura de 156 diapositivas» relaciona cada lámina con la sección de lectura pertinente.

14. Bibliografía, videos y lecturas ampliatorias

Las siguientes fuentes sustentan el apunte. Los diagramas de la plataforma de reservas son **ejemplos didácticos**, no resultados de investigaciones reales. En los estudios metodológicos se recomienda consultar los artículos completos antes de aplicar una técnica en un proyecto profesional.

Fuente y ampliación: 01. Olga Carreras · Arquitectura de información: fundamentos
Fuente y ampliación: 02. NN/g · Information Architecture: Study Guide
Fuente y ampliación: 03. NN/g · Flat vs. Deep Website Hierarchies
Fuente y ampliación: 04. NN/g · Card Sorting: Uncover Users' Mental Models
Fuente y ampliación: 05. NN/g · Tree Testing: Evaluate Menu Labels and Categories
Fuente y ampliación: 06. NN/g · Information Scent
Fuente y ampliación: 07. NN/g · Navigation: You Are Here
Fuente y ampliación: 08. NN/g · Menu Design: Checklist
Fuente y ampliación: 09. NN/g · How to Interpret Dendrograms (video/guía)
Fuente y ampliación: 10. NN/g · Hidden Navigation and Hamburger Menus
Fuente y ampliación: 11. Rosenfeld, Morville y Arango · Information Architecture (4th ed.)
Fuente y ampliación: 12. Pérez-Montoro · Arquitectura de la información en entornos web (referencia bibliográfica)