Matriz Maestra de Integración (SoT + ACL + Eventos)

1) Propósito y alcance

Este documento define la **“tabla maestra de integración”** lista para ingeniería/QA, incluyendo:

- **SoT (Source of Truth)** por entidad (quién es el dueño del dato).
- **ACL** (autorización) y reglas mínimas de seguridad/privacidad por recurso.
- **Eventos** (catálogo, productores/consumidores, contrato mínimo).
- **IDs, correlación, idempotencia, ordering, retries y DLQ**.
- **Contratos API mínimos** por SoT para backfill, reconciliación y lectura puntual.

Módulos cubiertos (según el set actual de la plataforma): - Core Platform: IAM, Files, Notify, Audit, Analytics, Search, Integration Hub - CRM - Licitaciones - Información de Productos (PIM/Expediente) - Inventario/Compras - RRHH - Intranet - Comunicaciones - Gestor de Tareas - Canvas

---

2) Convenciones globales obligatorias

2.1 Identificadores canónicos

Todos los servicios y contratos deben usar estos IDs:

- `tenant_id` (ULID/UUID) — obligatorio en *todo*.
- `user_id` (ULID/UUID) — identidad digital (IAM).
- `person_id` (ULID/UUID) — persona organizacional (RRHH).
- `workspace_id` (ULID/UUID) — contenedor transversal (colaboración).
- `object_ref` — referencia universal a cualquier objeto:
- `object_type` (enum global)
- `object_id` (id del SoT)
- `tenant_id`

Regla: ningún módulo inventa IDs “locales” sin `tenant_id`. Todos los IDs son **globalmente únicos** dentro del tenant.

2.2 Correlación y trazabilidad

- `X-Request-Id` (por llamada HTTP)
- `correlation_id` (flujo cross-módulo; se propaga en eventos y llamadas internas)
- `causation_id` (evento/comando que causó el actual)

Regla QA: en un flujo E2E, debe ser posible reconstruir la cadena completa desde cualquier evento o API call usando `correlation_id`.

2.3 Idempotencia (comandos y eventos)

- Comandos HTTP **mutables** aceptan `Idempotency-Key`.

- Todos los eventos incluyen `idempotency_key` y `event_id`.
- Consumidores implementan **Inbox/Dedupe Store** (`(tenant_id, idempotency_key)` único) + procesamiento atómico.

Regla QA: reintentar N veces el mismo comando/evento no debe duplicar efectos (tareas duplicadas, ledger duplicado, notificaciones repetidas, etc.).

2.4 Concurrencia

- Recursos mutables expuestos por API usan `ETag` + `If-Match`.
- Eventos llevan `entity_version` (monótono por entidad) o `updated_at` fuerte.

Regla QA: simular edición concurrente debe producir (a) conflicto 409 o (b) merge controlado según el caso.

2.5 Clasificación de datos (privacidad)

Todos los eventos/recursos declaran: - `data_classification`: `public | internal | confidential | sensitive` - `pii_flags`: `none | pii_basic | pii_sensitive | financial | health`

Regla: RRHH y cualquier dato personal sensible debe minimizar payload (ver sección de eventos).

---

3) Modelo ACL unificado (RBAC + ABAC + ACL por recurso)

3.1 Principals

- `user`
- `group`
- `service_account`

3.2 Scopes

- `tenant`
- `org_unit`
- `workspace`
- `resource`

3.3 Acciones canónicas

`create, read, update, delete, approve, export, share, admin`

3.4 ABAC mínimo

Atributos mínimos para decisión: - `owner_user_id` - `org_unit_id` / `cost_center_id` - `workspace_id` - `visibility`: `private | internal | public_link | external_shared` - `classification`

3.5 Herencia por contenedor

- Tareas: herencia por Space/Folder/List.
- Canvas: herencia por board y parent_board.
- Intranet: herencia por área/espacio editorial.
- Comunicaciones: por conversación + membership.

Regla QA: un cambio de permisos en un contenedor debe reflejarse en (a) acceso API, (b) búsquedas (security trimming) y (c) UI.

---

4) Tabla maestra SoT + ACL + Eventos (por entidad)

Formato de columnas: - **Entidad**: nombre canónico - **SoT**: módulo dueño - **PK**: clave primaria - **Natural Key / External Key**: claves de dedupe/reconciliación - **ACL Scope**: dónde se aplica la política - **Clasificación**: sensibilidad - **Eventos**: lista mínima (producer) - **Consumidores**: módulos típicos - **Backfill API**: endpoints mínimos para reconstrucción

4.1 Core Platform (obligatorio)

Entidad	SoT	PK	Natural/External	ACL Scope	Clasificación	Eventos (p
Tenant	Core-IAM	tenant_id	dominio/slug	tenant	internal	core.tenan
User	Core-IAM	user_id	email	tenant	internal (PII basic)	core.user.c deactivate
Group	Core-IAM	group_id	nombre+scope	tenant/ workspace	internal	core.group deleted
Role	Core-IAM	role_id	nombre+scope	tenant	internal	core.role.c deleted
RoleAssignment	Core-IAM	role_assignment_id	(principal,role,scope)	tenant/ workspace/ resource	internal	core.role_a
File	Core-Files	file_id	hash_sha256 + size	resource	confidential	core.file.cr deleted
Notification	Core-Notify	notification_id	(user,type,ref,ts)	user	internal	core.notifi
AuditRecord	Core-Audit	audit_id	(ref,ts,actor)	tenant/ resource	confidential	core.audit

Entidad	SoT	PK	Natural/External	ACL Scope	Clasificación	Eventos (p
AnalyticsEvent	Core-Analytics	analytics_event_id	(type,ref,ts)	tenant	internal	core.analy
SearchIndexDoc	Core-Search	search_doc_id	object_ref	tenant	internal	core.searc

Notas: - Si temporalmente File/Notify/Audit no se centraliza, se debe definir un **SoT provisional** (p.ej., Comms) y mantener contrato compatible para migración.

4.2 RRHH

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos
Person	RRHH	person_id	RUT/ Pasaporte (si aplica)	tenant/ org_unit	sensitive (PII)	hr.person.created/ updated
Engagement/ Contract	RRHH	engagement_id	(person_id, tipo, fechas)	tenant/ org_unit	sensitive	hr.engagement.start ended/updated
OrgUnit	RRHH	org_unit_id	código	tenant	internal	hr.orgunit.created/ updated
ManagerRelation	RRHH	mgr_rel_id	(person_id, manager_id)	tenant	sensitive	hr.manager.changed
HRDocument	RRHH	hr_doc_id	(person_id, doc_type, ts)	resource	sensitive	hr.document.created signed
PayrollRun	RRHH	payroll_run_id	(periodo, company)	tenant/ company	sensitive(financial)	hr.payroll.closed

ACL RRHH (regla mínima): - `hr_admin`: CRUD datos base (no remuneraciones) - `payroll_admin`: acceso remuneraciones - `legal_auditor`: read/export con masking - `manager`: read limitado a su equipo (ABAC)

4.3 Intranet

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos
Page	Intranet	page_id	slug+area	workspace/ area	internal	it.page.published/ updated/archived
NewsPost	Intranet	news_id	ts+area	workspace/ area	internal	it.news.published/ updated
Document	Intranet	doc_id	(title,version)	workspace/ area	confidential	it.document.published/ versioned
KBArticle	Intranet	kb_id	slug	workspace/ area	internal	it.kb.published/ updated
AudienceRule	Intranet	audience_id	nombre	tenant	internal	it.audience.updated
ServiceRequest	Intranet	req_id	(requester, ts)	tenant	internal	it.request.created/ approved/fulfilled/ closed

ACL Intranet: - `editor`: create/update draft - `reviewer`: approve - `publisher`: publish - `reader`: read

4.4 Comunicaciones

Entidad	SoT	PK	Natural/External	ACL Scope	Clasificación	Eventos
Conversation	Comms	conversation_id	(workspace_id, name, type)	workspace	internal	comms.conversatio archived
Membership	Comms	membership_id	(conversation_id, user_id)	workspace	internal	comms.membershi removed
Message	Comms	message_id	client_message_id (idempotencia)	conversation	internal/ confidential	comms.message.p edited/deleted
File (si no hay Core-Files)	Comms	file_id	hash+size	resource	confidential	comms.file.uploade

Entidad	SoT	PK	Natural/External	ACL Scope	Clasificación	Eventos
Notification (si no hay Core-Notify)	Comms	notification_id	(user,type,ref,ts)	user	internal	comms.notification read

ACL Comms: - `owner/moderator/member/guest` por conversación. - Acciones: `post`, `edit_own`, `delete_own`, `moderate`, `invite`, `archive`.

4.5 Gestor de Tareas

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos	Consumidor
Workspace/ Space	Tasks	space_id	(workspace_id, name)	workspace	internal	tasks.space.created/ updated	BI
Folder/List	Tasks	list_id	(space_id, name)	workspace	internal	tasks.list.created/ updated	BI
Task	Tasks	task_id	(list_id, title)	resource	internal	tasks.task.created/ updated/ status_changed/ deleted	notify, BI, CRM links
View	Tasks	view_id	(scope, type, name)	workspace	internal	tasks.view.created/ updated/shared	—

ACL Tasks: - Herencia por contenedor: Space/Folder/List. - Roles: `admin`, `member`, `guest`. - Acciones: CRUD task según rol.

4.6 Canvas

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos
Board	Canvas	board_id	(workspace_id, name)	workspace/ board	internal	canvas.board.created/ updated/archived
Element	Canvas	element_id	(board_id, type, created_at)	board	internal	canvas.element.added updated/moved/delete

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos
CommentThread/ Comment	Canvas	comment_id	(thread_id, ts)	board	internal	canvas.comment.adde resolved
ShareLink	Canvas	share_link_id	token	resource	confidential	canvas.share_link.crea revoked
ExportJob	Canvas	export_job_id	(board_id, ts)	board	confidential	canvas.export.complet failed

ACL Canvas: - Roles por board: `owner`, `editor`, `commenter`, `viewer`. - `public_link` solo si se habilita explícitamente (ShareLink).

4.7 CRM

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos	Consumidores
Company	CRM	company_id	RUT/ Domain/ Name (dedupe)	tenant	internal	crm.company.created/ updated/merged	BI
Contact	CRM	contact_id	email/phone (dedupe)	tenant	internal(PII)	crm.contact.created/ updated/merged	BI
Deal	CRM	deal_id	(company_id, title)	tenant	internal	crm.deal.created/ updated/ stage_changed/closed	BI, tasks automation
Ticket	CRM	ticket_id	(company/ contact, ts)	tenant	internal	crm.ticket.created/ updated/closed	BI, postventa (futuro)
Activity	CRM	activity_id	(ref, ts, type)	tenant	internal	crm.activity.created	BI

ACL CRM: - `sales_rep`: CRUD en cuentas asignadas, create deals, update stage. - `sales_manager`: export, override, reasignar.

4.8 Licitaciones

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos
RawTender/ RawOC	Licitaciones	source_id	MercadoPublico ID (external)	tenant	internal	lic.raw.ingested
Opportunity	Licitaciones	opportunity_id	(source_id, tenant)	tenant	internal	lic.opportunity.created/ updated/ stage_changed
Rule	Licitaciones	rule_id	nombre	tenant	internal	lic.rule.created/ updated
Alert	Licitaciones	alert_id	(opportunity, type, ts)	tenant	internal	lic.alert.created

ACL Licitaciones: - `bid_manager`: manage rules, pipeline. - `bid_contributor`: read/update oportunidades.

4.9 Inventario/Compras

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos	C
Supplier	Inventory	supplier_id	RUT/Name	tenant	internal	inv.supplier.created/ updated	E
SKU/Item	Inventory	sku_id	supplier_sku	tenant	internal	inv.sku.created/ updated	E r
Warehouse/Bin	Inventory	wh_id	code	tenant	internal	inv.warehouse.created/ updated	-
PR	Inventory	pr_id	(requester, ts)	tenant/ org_unit	internal	inv.pr.created/ submitted/approved/ rejected	E
RFQ/Quote	Inventory	rfq_id	(pr_id, supplier)	tenant	internal	inv.rfq.sent; inv.quote.received	E
PO	Inventory	po_id	(supplier, ts)	tenant	internal	inv.po.created/ approved/sent/ cancelled	f (
GRN	Inventory	grn_id	(po_id, ts)	tenant	internal	inv.grn.posted	f (

Entidad	SoT	PK	Natural/ External	ACL Scope	Clasificación	Eventos	C
StockLedgerEntry	Inventory	ile_id	(sku, wh, ts, ref)	tenant	internal	inv.stock_ledger.posted	E

ACL Inventory: - `buyer`: create PR/RFQ/PO - `approver`: approve PR/PO - `warehouse_op`: post GRN, stock moves - ABAC: warehouse scope por rol/ubicación.

4.10 Información de Productos (PIM)

Entidad	SoT	PK	Natural/External	ACL Scope	Clasificación	Eventos
Product	PIM	product_id	fabricante+modelo	tenant	internal	pim.product.cre lifecycle_chang
Asset	PIM	asset_id	(product_id, kind)	resource	confidential	pim.asset.creat
AssetVersion	PIM	asset_version_id	(asset_id, version)	resource	confidential	pim.asset_versi obsolete
Requirement	PIM	requirement_id	(product_id, dimension)	tenant	internal	pim.requiremer
CompletenessSnapshot	PIM	snapshot_id	(product_id, ts)	tenant	internal	pim.completen

ACL PIM: - `product_owner`: CRUD producto, aprobar versión. - `contributor`: proponer assets. - `viewer`: read.

5) Tabla de enlaces cross-módulo (EntityLink / ExternalRef)

5.1 ExternalRef (obligatoria)

Para toda integración con sistemas externos (Mercado Público, ERP, firma, etc.):

- `external_system` (enum)
- `external_id`

- `subject_object_ref`
- `metadata`

Invariante QA: `(tenant_id, external_system, external_id)` es único.

5.2 EntityLink (obligatoria)

Relaciones entre entidades sin acoplar schemas:

- `link_id`
- `from_object_ref`
- `to_object_ref`
- `relation_type` (`RELATED_TO`, `DERIVED_FROM`, `ATTACHED_TO`, `DISCUSSION_OF`, `WORK_OF`, `BLOCKS`, etc.)

Ejemplos obligatorios de links: - `lic.opportunity` → `crm.deal` (DERIVED_FROM) - `crm.deal` → `comms.conversation` (DISCUSSION_OF) - `pim.requirement` → `tasks.task` (WORK_OF) - `canvas.board` → `lic.opportunity` (RELATED_TO) - `inv.po` → `lic.opportunity` (RELATED_TO) cuando la compra es por licitación

---

6) Checklist QA por eventos (idempotencia, ordering, retries, DLQ)

6.1 Checklist general (aplica a TODO evento)

1) **Envelope válido**: incluye `tenant_id`, `event_id`, `event_type`, `event_version`, `occurred_at`, `correlation_id`, `idempotency_key`. 2) **Clasificación correcta**: `data_classification` y `pii_flags` coherentes. 3) **Payload mínimo**: sin campos innecesarios, especialmente en RRHH. 4) **Idempotencia**: re-procesar evento no duplica efectos. 5) **Persistencia outbox**: publicación es atómica con la transacción del SoT. 6) **Retries**: backoff exponencial; máximo N; luego DLQ. 7) **DLQ**: contiene motivo + payload + estado. 8) **Schema evolution**: consumidores toleran campos extra; cambios breaking suben `event_version`. 9) **Ordering**: - Por defecto: ordering **solo garantizado por entidad** (clave de partición = `tenant_id + object_type + object_id`). - Consumidores deben tolerar out-of-order global. 10) **Observabilidad**: logs/traces con `correlation_id`.

6.2 Checklist específico por tipo

- **Created**: la entidad debe ser GET-able por API dentro de SLA (read-your-writes si aplica).
- **Updated**: `entity_version` incrementa; cambios parciales deben estar claros.
- **Deleted**: soft-delete preferido; evento incluye `deleted_at`.
- **StateChanged** (PO approved, deal stage changed):
- validación de transición de estado
- evento incluye `from_state`, `to_state`, `reason_code`.

6.3 Pruebas de resiliencia (mínimo obligatorio)

- Duplicados: publicar el mismo evento 10 veces.
- Out-of-order: enviar `updated(v=3)` antes que `updated(v=2)`.

- Retry con fallo temporal del consumidor.
- DLQ: forzar payload inválido.

---

7) Contratos API mínimos por SoT (para backfills/reconciliación)

Objetivo: poder reconstruir read-models y validar consistencia sin acceder a DBs internas.

7.1 Patrón mínimo por servicio

Cada SoT debe exponer: - **GetById**: `GET /{domain}/{entity}/{id}` - **ListUpdatedSince**: `GET /{domain}/{entity}?updated_since=...&page_token=...` - **GetByExternalRef** (si aplica): `GET /{domain}/{entity}?external_system=X&external_id=Y` - **BulkGet** (para integraciones): `POST /{domain}/{entity}/bulk-get` (ids[]) - **Health/Version**: `GET /health`, `GET /version` (commit, schema)

7.2 APIs mínimas por módulo

Core-IAM

- GET /users/{id}
- GET /users?updated_since=
- GET /groups/{id}
- GET /role-assignments?principal_id=

CRM

- GET /crm/companies/{id}
- GET /crm/contacts/{id}
- GET /crm/deals/{id}
- GET /crm/deals?updated_since=
- GET /crm/activities?ref_object_type=&ref_object_id=

Licitaciones

- GET /lic/opportunities/{id}
- GET /lic/opportunities?updated_since=
- GET /lic/raw/{id}
- GET /lic/alerts?opportunity_id=

PIM

- GET /pim/products/{id}
- GET /pim/products?updated_since=
- GET /pim/assets/{id}
- GET /pim/requirements?product_id=

Inventario

- GET /inv/pos/{id}
- GET /inv/pos?updated_since=

- GET /inv/grns/{id}
- GET /inv/ledger?updated_since=
- GET /inv/skus/{id}

RRHH

- GET /hr/persons/{id}
- GET /hr/persons?updated_since=
- GET /hr/orgunits?updated_since=
- GET /hr/engagements/{id}

Intranet

- GET /intranet/pages/{id}
- GET /intranet/pages?updated_since=
- GET /intranet/docs/{id}
- GET /intranet/news?updated_since=

Comms

- GET /comms/conversations/{id}
- GET /comms/conversations/{id}/members
- GET /comms/conversations/{id}/messages?since=

Tasks

- GET /tasks/{id}
- GET /tasks?updated_since=
- GET /tasks/views/{id}

Canvas

- GET /canvas/boards/{id}
- GET /canvas/boards/{id}/elements?since=
- GET /canvas/exports/{id}

---

8) Reconciliación y backfill (procedimientos QA/Operación)

8.1 Reconciliación por evento vs API

Para cada entidad, QA debe validar: - Conteo por periodo (eventos vs `ListUpdatedSince`). - Último `entity_version` por objeto coincide. - Tolerancia a gaps: backfill rellena missing.

8.2 Backfill job estándar

- Entrada: `entity_type`, `updated_since`, `updated_until`, `tenant_id`.
- Proceso: 1) Llamar `ListUpdatedSince` al SoT. 2) Reindexar Search/ReadModels. 3) Registrar Audit técnico (quién corrió, rango, resultados).

8.3 Invariantes de coherencia (mínimos)

- Toda `Task` con `context_ref` debe apuntar a un objeto existente (resoluble por API).
- Toda `Opportunity` con link a `Deal` debe tener `ExternalRef` a Mercado Público.
- Todo `StockLedgerEntry` debe referenciar un documento (`ref_doc`) existente (PO/GRN/ajuste).
- Todo `ShareLink` debe tener expiración/estado consistente.

---

9) Plan de pruebas E2E (mínimo ejecutable por QA)

9.1 Licitaciones → CRM + Tasks + Comms

1) Ingesta raw → creación de Opportunity. 2) "Trabajar oportunidad": crea Deal + Conversation + set de Tasks. 3) Cambiar stage Opportunity y verificar stage Deal (si se sincroniza). 4) Verificar notificaciones por menciones y vencimientos.

9.2 PIM completitud → Task

1) Crear Product. 2) Detectar requirement faltante. 3) Emitir evento missing_detected. 4) Generar Task con link y verificar trazabilidad.

9.3 Inventario PO → GRN → Ledger

1) Crear PR → aprobar → PO. 2) Post GRN. 3) Ledger posted (inmutable) y stock actualizado. 4) Reintento del mismo GRN/ledger no duplica movimiento.

9.4 RRHH onboarding → IAM + Intranet directory

1) Crear Person/Engagement. 2) Evento crea User (o vínculo) y aparece en directorio. 3) Offboarding: desactiva acceso, reasigna ownership.

9.5 Canvas board → links → export

1) Crear board y elementos. 2) Crear share link (si habilitado) y validar permisos. 3) Ejecutar export y verificar output file_id.

---

10) "Definition of Done" de integración (criterios de aceptación)

Un módulo se considera **integrado** cuando cumple: 1) Tabla SoT completada (entidades, keys, clasificación). 2) ACL implementada y probada (incluye herencia y ABAC mínimo). 3) Outbox + publicación de eventos + consumidores idempotentes. 4) APIs mínimas de backfill disponibles. 5) Search trimming correcto (si indexa). 6) Observabilidad: logs + métricas + trazas con `correlation_id`. 7) Suite QA: duplicados, out-of-order, retries, DLQ y E2E.

---

11) Apéndice: Enum global sugerido para object_type

```
TENANT, USER, PERSON, ORG_UNIT, COMPANY, CONTACT, DEAL, TICKET, ACTIVITY, LIC_RAW, LIC_OPPORTUNITY, PRODUCT, PRODUCT_ASSET, SKU, SUPPLIER, PR, RFQ, PO, GRN, LEDGER_ENTRY, TASK, TASK_VIEW, CONVERSATION, MESSAGE, BOARD, ELEMENT, INTRANET_PAGE, INTRANET_DOC, KB_ARTICLE, SHARE_LINK, EXPORT_JOB, FILE, NOTIFICATION
```

Notas finales

- Esta matriz es el contrato de integración “de verdad”. Cualquier cambio debe pasar por control de versión (PR + review de arquitectura + actualización de QA cases).
- Cuando se incorporen módulos futuros (Comercial completo, Postventa, Proyectos, Documentos/Contratos, Finanzas externo), se agregan sus entidades en esta misma tabla y se conectan vía `ExternalRef` + `EntityLink` sin romper los consumidores existentes.