Plano — completar o mapeamento de campos INDE → PDA

Contexto

O plugin `ckanext-cgu-inde` importa metadados geoespaciais (ISO 19139 / perfil MGB) do catálogo da INDE para o Portal de Dados Abertos. Hoje mapeia apenas parte dos campos: título, descrição, organização, e-mail, palavras-chave, periodicidade, idioma, cobertura temporal e linhagem.

A gestão definiu a correspondência completa numa planilha oficial (`Correspondências campos Dados-INDE.xlsx`, nesta mesma pasta, não versionada). Este plano executa as lacunas **em etapas pequenas, cada uma testável e commitável isoladamente**.

Risco de conformidade hoje: `observanciaLegal` recebe o valor fixo `"1"` (Público) para todo conjunto, independentemente da restrição declarada no metadado.

Decisões já tomadas

Tema	Decisão
Ordem	Conformidade primeiro
Visibilidade	Importar tudo como privado, conforme a planilha
Duplicidade	Implementar como na planilha: duplicado não-INDE → privado + e-mail aos gestores

Descobertas que orientam o plano

1. O vocabulário do PDA já existe em `ckanext-cgu-dcat-br/.../vocabulary.py`: `_THEME_LIST` (32 temas), `_FREQUENCY_LIST`, `_LANG_LIST` (pt,en,es,fr,de,it,ja,zh), `_RESOURCE_TYPE_LIST`, `_SPATIAL_LIST`, `_LICENSE_MAP`, observância legal = dígito 1..34.

2. ⚠ Deixar `observanciaLegal` em branco publica como PÚBLICO. Em `cgu_dcat_br/profiles.py:106`, a ausência do extra vira `dct:accessRights = SEI/1`. E um valor não-numérico (ex.: "Restrito (INDE)") faz o campo sumir do RDF. **As duas saídas "seguras" da planilha são inseguras no PDA** — daí a decisão de marcar `private=True` + extra de revisão quando a restrição for indeterminada.

3. ⚠ `useConstraints` NÃO é extraído pelo ckanext-spatial. O elemento `use-constraints` aponta para `useLimitation` (texto livre), não para o `MD_RestrictionCode`. Só `access-constraints` traz código, e só de `accessConstraints`. **Observância legal e licença exigem um parser complementar** — não são "só uma tabela no JSON".

4. ⚠ Escrever `groups` REMOVE os grupos ausentes da lista (CKAN `model_save.py:225`). Como o harvest reimporta a cada rodada, isso apagaria curadoria manual do portal. Precisa fazer merge com os grupos atuais.

5. O tema "Geoespacial" não existe — nem como grupo (27 grupos em homologação e produção, sem `geoespacial`) nem no vocabulário do PDA. Mas `geografia` **existe nos dois**. Usar `geografia` evita depender da criação de um tema novo.

6. Os campos de restrição da INDE são muito mais pobres que a planilha supõe. Amostra real: `access-constraints` quase sempre vazio (quando presente: `licenceUnrestricted`, `semRestricoes`); `otherConstraints` em texto livre português. A conversão precisa casar por contenção e sem diferenciar caixa, e a maioria dos conjuntos ficará sem valor — resultado legítimo, a reportar à gestão.

7. Recursos são abundantes, formato declarado não. 37 recursos amostrados: `protocol` bem preenchido (WWW:LINK, OGC:WMS, WWW:DOWNLOAD), `name` em 26/37, mas `MD_Format.name` **vazio em 100%**.

8. O CKAN já normaliza formato de recurso (`clean_format`): `application/pdf` →PDF, `wms` →WMS. A etapa de formato vira só ordem de prioridade + fallback.

9. A planilha contradiz a si mesma em dois pontos, e o código atual está certo: - Periodicidade: manda `continual` →DIARIA, `irregular` →OUTRAS, `biennially` →"maior que anual", alegando que não existem no PDA. **Existem** (`CONTINUO` , `IRREGULAR` , `BIENAL`). Manter o código e reportar. - Temas: a aba principal espera 11 temas para `planningCadastre; society` , mas a aba "Temas" marca os dois como "sem correspondência". Implementar a aba tabular.

10. ⚠ O `try/except` guarda-chuva em `get_package_dict` (plugin.py:291-300) importa o conjunto **sem nenhuma adaptação do PDA** se qualquer regra falhar. Com 8 regras novas, uma exceção em "cobertura espacial" derrubaria título, órgão e observância legal junto. Cada regra nova precisa de `try/except` próprio.

11. O ambiente local não tem nenhum tema cadastrado (0 grupos) — a etapa de temas exige semear os grupos localmente antes de testar.

Pré-requisitos externos

Pré-requisito	Responsável	Bloqueia
Definir destinatários do e-mail de duplicidade	Gestão	Etapa 14
Assinar as divergências da planilha (descoberta 9)	Gestão	Etapa 7 (só relato)
(opcional) Criar tema `geoespacial` no PDA	Gestão	— usar `geografia` enquanto isso

Estratégia técnica

Tabelas de conversão: expandir `data/field_mapping.json` (um arquivo = uma planilha), com uma exceção — a lista de municípios do IBGE (~120 KB) vai em `data/br_municipios.json` separado, carregado sob demanda.

Reuso do vocabulário: criar `vocabulary.py` no cgu_inde como **adaptador com import preguiçoso e fallback permissivo** das funções públicas do cgu-dcat-br (`get_theme_uri` , `get_frequency_uri` , ...). Nunca copiar as listas, nunca importar nomes privados. O acoplamento fica no **teste** (`test_vocabulary_alinhado.py` , com `pytest.importorskip`), que falha se alguém mudar o vocabulário do PDA.

Módulos novos (mapping.py tem 184 linhas e receberia ~500): `vocabulary.py` , `iso_extra.py` , `constraints.py` , `themes.py` , `resources.py` , `coverage.py` , `duplicates.py` — cada um com funções puras e teste próprio.

Parser complementar (`iso_extra.py`): o `ISODocument` do ckanext-spatial está em site-packages (instalado por pip, não vendorizado) e é importado diretamente pelo harvester — não é plugável. Re-parsear com lxml custa ~1 ms/registro e **já é padrão neste código** (`_early_skip_reason` faz isso). Preferível a um patch de imagem, que exigiria rebuild a cada iteração e não seria testável com pytest.

Etapas — visão geral

#	Etapa	Bloco	Retorno	Observação
0	Whitelist derivada + cache + adaptador de vocabulário	Fundação	—	Elimina a armadilha do `_PDA_EXTRA_KEYS` antes de criar 6 extras
1	CLI `cgu-inde previa` (dry-run de 1 registro)	Fundação	alto	Maior ganho de velocidade do plano
2	Cobertura temporal `dd/mm/aaaa` (bug)	Conformidade	corrige RDF	Esqueleto ambulante do fluxo de teste
3	`iso_extra.py` — XPaths que faltam	Conformidade	destrava 4/5/13	Maior risco técnico, isolado
4	**Observância legal**	Conformidade	**conformidade**	O risco de hoje
5	Licença de uso	Conformidade	baixo (ver desc. 6)	
6	Visibilidade (`private`)	Conformidade	decisão da gestão	Muda os 44 já importados

#	Etapa	Bloco	Retorno	Observação
7	Idiomas + periodicidade	Completude	pequeno	Só JSON + teste
8	URL de origem + campo Origem	Completude	rastreabilidade	
9	Área técnica responsável	Completude	médio	
10	**Temas**	Completude	**alto** (descoberta)	Merge obrigatório (desc. 4)
11	Tipo do recurso	Completude	alto	Ordem das regras ≠ ordem da planilha
12	Formato do recurso	Completude	**quase nulo**	Recomendo não fazer (desc. 7 e 8)
13	Cobertura espacial	Completude	médio	Municípios são sub-etapa opcional
14	Duplicidade: privado + e-mail	Governança	médio	E-mail fora do harvest
15	Documentação e fechamento	Governança	—	

Detalhamento das etapas

Bloco A — Fundação

Etapa 0 — Whitelist derivada, cache e adaptador de vocabulário. `load_json_data` ganha `lru_cache`; `field_mapping.json` ganha `pda_extra_keys`; `mapping._pda_extra_keys()` deriva o conjunto do JSON, substituindo a constante `_PDA_EXTRA_KEYS`. A partir daqui, criar extra novo é editar o JSON. Aceite: `run-test inde-ifpi` produz **diff vazio** nos extras dos 5 conjuntos.

Etapa 1 — `ckan cgu-inde previa --uuid <guid> [--xml] [--json]`. Busca um registro, roda todo o mapeamento e imprime o `package_dict` **sem escrever no banco**. Torna as 13 etapas seguintes testáveis em segundos. Commitar 4-6 XMLs reais como fixtures. Aceite: contagem de `package_search` idêntica antes/depois; saída bate com o `package_show` do mesmo conjunto.

Bloco B — Conformidade

Etapa 2 — Cobertura temporal. `utils.format_pda_date()` aceitando `AAAA-MM-DD`, ISO com hora, `AAAA-MM` e `AAAA`, devolvendo `dd/mm/aaaa`. Aceite: `dct:temporal` passa a aparecer no RDF (hoje some silenciosamente).

Etapa 3 — `iso_extra.py`. Extrai `use-constraints-code`, `other-constraints`, `edition`, `geographic-description`, `resource-mime`, `distribution-format`. Integrado em `get_package_dict` antes das regras, com `try/except`. Já entrega "Versão" (`package_dict['version']`). Aceite: fixtures devolvem os códigos corretos; ICMBio importa 16/16 sem erro.

Etapa 4 — Observância legal. `constraints.observancia_legal()` devolvendo `(valor, motivo_de_revisão)`. Prioridade: `restricted` > `patent` / `trademark` (→9) > `intellectualPropertyRights` (→4) > `copyright` (→2) > `license` / `licence` (→1 **só se** houver licença aberta em otherConstraints) > `otherRestrictions` > desconhecido. Restrição indeterminada → `private=True` + extra `inde_revisar` (descoberta 2). Remove `observanciaLegal` dos `defaults`. Comando `cgu-inde auditar-observancia`. Aceite: 10/10 linhas da planilha cobertas por teste; nenhum conjunto vira `1` sem licença aberta declarada; RDF de conjunto restrito não contém `SEI/1`.

Etapa 5 — Licença. `constraints.licenca()` casando as 12 grafias da planilha. **Teste obrigatório**: `CC BY-SA` / `BY-NC` / `BY-ND` **não** podem virar `cc-by`. Aceite: 12 linhas passam; 4 casos negativos rejeitados.

Etapa 6 — Visibilidade. `ckanext.cgu_inde.private_datasets`, **padrão ligado** (decisão da gestão), com override por fonte. Regras de negócio (restrição indeterminada, duplicado) sempre vencem em favor de privado. Aceite: conjuntos não aparecem em busca anônima. ⚠ Avisar a equipe: os 44 já importados em homologação ficarão privados no próximo harvest.

Bloco C — Completude

Etapa 7 — Idiomas e periodicidade. `deu→de`, `ita→it`, `chi→zh` (a planilha diz "ch", que não existe no ISO 639-1); `bimestral→BIMESTRAL`, `periodic→OUTRAS`, `biennially→BIENAL`, `quinquennal→OUTRAS`. Registrar as divergências da descoberta 9.

Etapa 8 — URL de origem. Extra `urlOrigem` montado do UUID (só se o guid tiver forma de UUID) + extra `origem = "INDE"`.

Etapa 9 — Área técnica. `individual-name` (fallback `position-name`) → `maintainer`, mantendo o nome da organização quando ausente.

Etapa 10 — Temas. `topic_map` com as 19 linhas da aba (`utilitiesCommunication` gera 2 temas); `always: ["geografia"]`. **Merge obrigatório** com os grupos atuais (descoberta 4). Resolver grupos com **um** `group_list` por processo. Comando `cgu-inde temas --checar` para diagnosticar grupos faltantes antes de ligar. Aceite: todo conjunto com ≥1 grupo; `dcat:theme` no RDF; grupo manual sobrevive a um segundo harvest.

Etapa 11 — Tipo do recurso. Ordem de avaliação **diferente da ordem das linhas da planilha**: dicionário → documentação → API → dados → outro. Teste de regressão: o exemplo da própria planilha (`.../DICIONARIO_DADOS/....pdf` com protocolo de download) deve dar "Dicionário de dados", não "Dados".

Etapa 12 — Formato do recurso (OPCIONAL). Recomendo **não implementar**: `MD_Format.name` está vazio em 100% da amostra e as demais prioridades já são o que `guess_resource_format()` + `clean_format` fazem. Só justifica se aparecer caso real.

Etapa 13 — Cobertura espacial. Brasil→FEDERAL, regiões→REGIAO_, UF→ESTADUAL, município→MUNICIPAL, país/continente→INTERNACIONAL. Texto desconhecido → em branco, não INTERNACIONAL* (classificar seria chute). Municípios (gazetteer IBGE) é sub-etapa opcional 13b. Ambiguidade "Rio de Janeiro" (UF e município): UF ganha.

Bloco D — Governança

Etapa 14 — Duplicidade. `dedup.veio_da_inde()` distingue pela presença de `inde_fonte` / `inde_uuid`. Duplicado INDE→INDE continua descartado; duplicado **não-INDE** entra privado com extra `inde_duplicado_de`. **Não enviar e-mail dentro do `get_package_dict`** — SMTP bloqueante ali derruba o fetch consumer. Seguir o padrão do `pending.py`: gravar em JSON e enviar por comando de cron `cgu-inde notifica-duplicados [--teste]`, espelhando `notify_organization` do cgu-harvester. Aceite: duplicado externo importado/privado/registrado; nenhum e-mail durante o harvest; `--teste` imprime sem enviar.

Etapa 15 — Documentação. Tabela final planilha × implementação com as divergências explicitadas para a gestão assinar; novas opções de `.ini`; comandos novos no README.

Fora de escopo

Item	Motivo
Granularidade espacial, versão anterior, substituído por, conjuntos associados, cobertura temporal do recurso, checksum, byteSize	A planilha manda deixar em branco
Criar o código "Restrito (INDE)" no vocabulário SEI	Depende do time do PDA; a decisão de privado+revisão cobre o risco
Remover o fallback `'1'` do `profiles.py`	Outra extensão, afeta todos os conjuntos do PDA. Issue separada
Alinhar periodicidade à planilha (desc. 9)	A planilha está errada; mudar pioraria o dado
Migração retroativa dos já importados	Sai de graça no próximo harvest; `auditar-observancia` mede antes/depois

Verificação

Unitários (sem banco):

```
docker exec ckan-cgu-ckan-1 bash -lc \
  'cd /app/extensions/ckanext-cgu-inde && source /app/venv/bin/activate && \
   python -m pytest ckanext/cgu_inde/tests/ -q --ckan-ini=$CKAN_CONFIG/ckan.ini'
```

Ponta a ponta local (http://localhost:8081):

```
docker exec ckan-cgu-ckan-1 bash -lc \
  'ckan -c $CKAN_CONFIG/ckan.ini cgu-inde setup-fonte --org <slug> --atualizar'
docker exec ckan-cgu-ckan-1 bash -lc \
  'ckan -c $CKAN_CONFIG/ckan.ini harvester run-test inde-<slug>'
```

Órgãos pequenos: `agencia-nacional-de-mineracao` (14), `instituto-federal-...-piaui` (5), `ministerio-do-desenvolvimento-e-assistencia-social-...` (4), `empresa-de-pesquisa-energetica-epe` (6 + 15 rejeitados), `instituto-chico-mendes-...-icmbio` (16).

O RDF é o teste real de que os valores estão no vocabulário — valor fora dele some sem erro: `curl -s localhost:8081/dataset/<nome>.rdf`.

Homologação: `run-test` **não funciona lá** (depende do pacote `factory`, ausente na imagem). Usar `harvester job inde-<slug>` (fila). Evitar IBGE (21.862 registros) e Embrapa (1.858): há um único processo de fetch para as ~94 fontes.

Histórico

Já implantado (commits `d9f1d56`, `aedf588`, `e3bd4a9`, `eaeb700`): correção da exclusão indevida de conjuntos, duplicação de organizações, reativação de conjuntos excluídos, CQL da EPE, os 25 órgãos do `org_setup.json`, resolução de organização com nome em maiúsculas, listagem completa de organizações e normalização de e-mail. Homologação alinhada com os slugs de produção; 44 conjuntos validados em 5 órgãos.