ADR-016: EInvoice Adapter Lifecycle and Contract

```
# ADR-016 — EInvoiceAdapter Lifecycle and Contract

**Status:** Accepted
**Date:** 2026-05-13
**Author:** Petter Graff (CodeCraft — Architecture Lead)
**Finverge Co-author:** Markos Zachariadis (Payments & Fiscal Integration)
**Decision-maker:** CEO Alem Baši?
**MC Task:** #100585 (Phase 0' ADR Consolidation — EInvoiceAdapter lifecycle)
**Supersedes:** ADR-016 v1 (2026-05-11, MC #100362) — this is the authoritative version
**Cross-references:**

- ADR-015 (CountryPlugin — `generateEInvoiceXml()` and `submitToFiscalPlatform()` delegate
to adapters)
- ADR-019 (Integration Adapter Registry — `AdapterConfig`, secret taxonomy, categories)
- ADR-023 §3.3 (backend country differentiation — market selected before adapter dispatch)
- `apps/api/src/main/kotlin/no/alai/bilko/einvoice/EInvoiceTypes.kt` (canonical types on
disk)
- `apps/api/src/main/kotlin/no/alai/bilko/country/hr/StorecoveHrFiskEInvoiceAdapter.kt` (HR
reference)
- Plan v3 §4b ADR-016 requirement + §4d HR critical path — `~/system/specs/bilko-multi-
market-architecture-plan-v3-2026-05-11.md`

---

## 1. Context

### 1.1 The Four-Platform Problem

Bilko targets four tax jurisdictions with four incompatible e-invoice fiscal platforms:

| Market | Platform                                | Transport  | Format                      |
Status          |
| ------ | --------------------------------------- | ---------- | --------------------------- |
--------------- |
| HR     | HR-FISK / FINA via Storecove            | Peppol AS4 | UBL 2.1 + HR CIUS           |
STUB (MC #8675) |
| RS     | SEF (efaktura.gov.rs)                   | REST API   | SEF XML (Serbian-specific) |
Phase 1S        |
| BA-FED | CPF (Centralna platforma za fakture) | TBD ~2027  | TBD                         |
Phase 1B        |
| BA-RS  | UINO (stub name)                        | TBD        | TBD                         |
Phase 1B        |

Without a canonical abstraction, each platform's integration detail bleeds into the core
invoice service — reproducing the Variant B coupling problem (ADR-bilko-002 §3).

### 1.2 Existing Types on Disk (verified 2026-05-11)

`EInvoiceTypes.kt` already defines (lines 1–224):

- `AdapterLifecycleState` enum: `STUB`, `SANDBOX_VERIFIED`, `PRODUCTION`
- `EInvoiceStatus` enum: `PENDING`, `APPROVED`, `REJECTED`, `CANCELLED`, `ERROR`
- `InvoiceTypeCode`: UNTDID codes 380, 381, 383, 384
- `Address`, `PartyInfo` / `Party` typealias
- `PaymentMeans`: `paymentMeansCode`, `paymentReference`, `iban`
- `TaxCategory` enum: `S, Z, E, K, G, O, AE` per EN 16931 BT-118
- `TaxBreakdown`, `InvoiceLine`, `CanonicalInvoice`, `SubmitResult`, `InvoiceTotals`
- `EInvoiceAdapter` interface with 4 methods + 2 properties

`AdapterTypes.kt` (in `no.alai.bilko.adapter`) defines:

- `AdapterErrorCode` enum with 10 codes including `NOT_IMPLEMENTED`
- `AdapterException(code, market, retryable, rawPayload, message, cause)`
```

The `EInvoiceAdapter` interface and lifecycle states exist but are not formally documented.
`StorecoveHrFiskEInvoiceAdapter` implements the interface — `serialize()` is fully operational
offline; all other methods throw `NOT_IMPLEMENTED`. This ADR formalises the contract and lifecycle governance.

---

2. Decision

2.1 EInvoiceAdapter Interface — Formal Contract

Defined in `apps/api/src/main/kotlin/no/alai/bilko/einvoice/EInvoiceTypes.kt` lines 200–224.
Reproduced here as the normative specification with full contract annotations:

```kotlin
interface EInvoiceAdapter {
    val jurisdiction: TaxJurisdiction
    val lifecycleState: AdapterLifecycleState

    /**
     * Serialize a canonical invoice to the adapter-specific wire format.
     *
     * CONTRACT:
     * - MUST be offline-capable — no network, no credentials required.
     * - MUST be deterministic: same [invoice] input produces identical bytes.
     * - MUST NOT log raw PII fields (OIB, IBAN, document_data) — call sanitizeForLog().
     * - Returns the full wire-format payload for the platform:
     *     HR: Storecove JSON envelope wrapping UBL 2.1 XML
     *     RS: SEF XML (Serbian Ministry of Finance schema)
     *     BA: CPF/UINO platform format (TBD)
     * - Throws AdapterException(VALIDATION_BUSINESS_RULE) for constraint violations
     *   (non-EUR currency for HR, invalid OIB, empty lines, etc.)
     * - AdapterConfig.enabled check is NOT performed here — callers check before invoking.
     * - This method is ALWAYS available, even in STUB lifecycle.
     */
    fun serialize(invoice: CanonicalInvoice): ByteArray

    /**
     * Submit the serialized invoice bytes to the fiscal platform.
     *
     * CONTRACT:
     * - Requires live credentials (API key, OAuth token, or certificate).
     * - MUST include an idempotency key (platform-specific — see §2.3).
     * - Returns SubmitResult on success; throws AdapterException on ALL failures.
     * - NEVER propagates platform-native exceptions (Ktor ResponseException, etc.) —
     *   map every platform exception to AdapterException before propagating.
     * - Implementations in STUB lifecycle MUST throw NOT_IMPLEMENTED (see §2.5).
     * - Idempotency: platforms may return 409 DUPLICATE on re-submission.
     *   Caller should treat 409 as success — extract submission ID from error body.
     *
     * @param serializedInvoice bytes from serialize()
     * @param invoice original CanonicalInvoice (needed for idempotency key generation)
     */
    fun submit(serializedInvoice: ByteArray, invoice: CanonicalInvoice): SubmitResult

    /**
     * Poll the fiscal platform for the current status of a submitted invoice.
     *
     * CONTRACT:
     * - [submissionId] is SubmitResult.platformInvoiceId from submit().
     * - Returns current EInvoiceStatus.
     * - This method is IDEMPOTENT — safe to call multiple times with the same submissionId.
     * - Callers implement exponential backoff; this method does NOT retry internally.
     * - Implementations in STUB lifecycle MUST throw NOT_IMPLEMENTED (see §2.5).
     * - NEVER log rawPayload without sanitizeForLog().
     */
    fun pollStatus(submissionId: String, invoice: CanonicalInvoice): EInvoiceStatus

    /**
     * Parse an inbound invoice from a raw fiscal platform webhook payload.
     *
     * CONTRACT:
     * - [rawPayload] is the raw bytes from the platform webhook (Storecove POST, SEF callback).
```

```
     * - Returns CanonicalInvoice with adapterMetadata populated for platform-specific fields:
     *     HR: "hr.supplierOib", "hr.buyerOib", "hr.pozivNaBroj"
     *     RS: "rs.supplierPib", "rs.buyerPib", "rs.sefId"
     * - Implementations in STUB lifecycle MUST throw NOT_IMPLEMENTED (see §2.5).
     * - NEVER log rawPayload before passing through sanitizeForLog().
     * - parseIncoming() is deferred for HR: not required for v1 HR GA (Phase 1H.6 scope).
     *   Implement 90 days post-GA (see Plan v3 §4d).
     */
    fun parseIncoming(rawPayload: ByteArray): CanonicalInvoice
}
```

2.2 CanonicalInvoice — EN 16931 Subset

The internal invoice representation, independent of any platform wire format. Defined in `EInvoiceTypes.kt` lines 141–156:

```kotlin
data class CanonicalInvoice(
    val id: String,                          // Internal UUID — Storecove document_id (D2 dedup)
    val invoiceNumber: String,               // BT-1: human-readable invoice number
    val issueDate: LocalDate,                // BT-2
    val dueDate: LocalDate,                  // BT-9
    val typeCode: InvoiceTypeCode,           // BT-3: UNTDID 1001 (380/381/383/384)
    val currencyCode: String,                // BT-5: ISO 4217 ("EUR", "RSD", "BAM")
    val jurisdiction: TaxJurisdiction,       // Routing discriminator (non-EN16931)
    val supplier: PartyInfo,                 // BG-4: name, taxId (OIB/PIB/JIB), address
    val buyer: PartyInfo,                    // BG-7: same structure
    val lines: List<InvoiceLine>,            // BG-25: quantity, unitPrice, lineTotal, taxRate
    val taxBreakdowns: List<TaxBreakdown>,   // BG-23: one entry per rate band
    val paymentMeans: PaymentMeans? = null,  // BG-16: paymentMeansCode, IBAN, reference
    val note: String? = null,                // BT-22: free text note
    val adapterMetadata: Map<String, String> = emptyMap(), // platform-specific extras
)
```

Field constraints:

Field	Constraint	Enforced in
`currencyCode`	"EUR" for HR (HALT-3 — Croatia adopted EUR 2023-01-01)	serialize() HR
`supplier.taxId`	OIB (HR, 11-digit ISO 7064 MOD 11,10) / PIB (RS, 9-digit) / JIB (BA, 13-digit)	serialize() per market
`lines`	Non-empty — EN 16931 §BG-25 minimum one line	serialize()
`taxBreakdowns`	Must sum to lines.(taxRate * lineTotal) — tolerance 0.01	InvoiceService
`adapterMetadata`	HR inbound: `hr.supplierOib`, `hr.buyerOib`, `hr.pozivNaBroj`	parseIncoming()

What CanonicalInvoice is NOT:

- Not a DB entity (mapped from `invoices` + `invoice_items` tables on read)
- Not a REST API DTO (API layer maps separately)
- Not versioned independently — evolves with EN 16931 minor revisions

2.3 Adapter Lifecycle State Machine

Defined in `EInvoiceTypes.kt` lines 22–26. Transition criteria formalised here:

```
STUB
  ? Compiles. All 3 network methods throw NOT_IMPLEMENTED.
  ? serialize() MAY be operational (HR: already works offline).
  ? AdapterConfig row not required.
  ?
  ? Transition criteria ? SANDBOX_VERIFIED:
  ?   1. Provider account provisioned (MC #8675 for HR/Storecove)
  ?   2. Credentials loaded in GCP Secret Manager (see §2.6 secret taxonomy)
```

```
  ?   3. 5 sandbox test invoice types pass with REAL platform submission IDs (§2.4)
  ?   4. pollStatus() confirmed for each submitted invoice
  ?   5. Proveo evidence file with submission IDs uploaded to BookStack
  ?   6. lifecycleState field updated to SANDBOX_VERIFIED in adapter source
  ?
  ?
SANDBOX_VERIFIED
  ? All 4 methods operational against provider sandbox.
  ? AdapterConfig(market, EINVOICE, enabled=true) in STAGE DB.
  ?
  ? Transition criteria ? PRODUCTION:
  ?   1. Securion audit: adapter error handling + PII sanitization (see §2.7)
  ?   2. 30 continuous days on STAGE Cloud Run with zero
  ?      AdapterErrorCode.PLATFORM_INTERNAL_ERROR alerts
  ?      (Prometheus metric: bilko_integration_request_total)
  ?   3. AdapterConfig(market, EINVOICE, enabled=true) in PRODUCTION DB
  ?   4. CEO sign-off (this is the go-live gate)
  ?
  ?
PRODUCTION
  ? Live. All 4 methods operational against production platform.
  ? Incident response: if critical error rate > 5% over 15min window,
  ?   automated alert ? Slack #bilko-incidents ? human decision to flip
  ?   AdapterConfig.enabled = false (no redeploy needed).
```

Current HR state (2026-05-13): STUB

- `serialize()`: WORKS (offline). Unit-tested.
- `submit()`: throws NOT_IMPLEMENTED — MC #8675 pending
- `pollStatus()`: throws NOT_IMPLEMENTED
- `parseIncoming()`: throws NOT_IMPLEMENTED (deferred post-GA)

2.4 HR-FISK Storecove Sandbox Validation Matrix

5 invoice types required for SANDBOX_VERIFIED transition. All must produce real Storecove submission GUIDs (not mock strings). Proveo (Angie Jones) runs these tests.

#	Invoice Type	UNTDID Code	Scenario	Expected Storecove Response	Evidence Required
1	B2B outbound commercial	380	Supplier OIB + Buyer OIB both valid. EUR. 25% PDV. Standard commercial transaction.	HTTP 200 + `{"id": "<guid>", "status": "pending"}`	Storecove submission GUID in evidence file
2	B2G outbound	380	Buyer is HR government entity (OIB format same). `PaymentMeans.paymentMeansCode=30`.	HTTP 200 + GUID	GUID + Storecove routing.peppol.id verified as buyer OIB
3	Credit note	381	References original invoice number in `note` field. Negative line totals.	HTTP 200 + GUID	GUID + typeCode=381 confirmed in Storecove portal
4	Cancelled invoice	384	CORRECTIVE_INVOICE type. Status flow: submit ? pollStatus until APPROVED or REJECTED	HTTP 200 + GUID, then pollStatus APPROVED/REJECTED	GUID + final status confirmed
5	Inbound received	380 (inbound)	Storecove sends test webhook to Bilko's webhook endpoint. `parseIncoming()` invoked.	Webhook received. CanonicalInvoice returned. `hr.supplierOib` in adapterMetadata.	Log entry showing successful parse + extracted OIB value

HR-specific validation rules verified in each test case:

- `currencyCode = "EUR"` (HALT-3)
- Supplier OIB: ISO 7064 MOD 11,10 checksum valid
- Buyer OIB: ISO 7064 MOD 11,10 checksum valid
- CustomizationID: verify with Storecove support which to use (PEPPOL_BIS3 or HR_CIUS — TODO MC #8675 D3)
- `routing.peppol.scheme = "9934"` and `routing.peppol.id = <buyerOIB>`

Storecove-specific notes:

- Sandbox URL is the same as production (`api.storecove.com/api/v2`) — sandbox mode is a payload flag, not a different host. Set `STORECOVE_ENV=sandbox` env var.
- Idempotency key: SHA-256(`invoice.id` + `invoice.invoiceNumber`) ? sent as `Idempotency-Key` header.
  Platform returns HTTP 409 on duplicate — treat as success (re-fetch GUID from error body).
- `document_id` field in Storecove payload = `CanonicalInvoice.id` (Bilko UUID) — Storecove dedup key, prevents double-billing on retry (D2 in StorecoveHrFiskEInvoiceAdapter).

2.5 NOT_IMPLEMENTED Transition Rules

`AdapterErrorCode.NOT_IMPLEMENTED` is the canonical error code for STUB lifecycle methods. Rules for callers and implementers:

Implementer rules:

1. Any STUB lifecycle method that is not yet operational MUST throw:
   ```kotlin
   throw AdapterException(
       code = AdapterErrorCode.NOT_IMPLEMENTED,
       market = jurisdiction,
       retryable = false,
       rawPayload = "",
       message = "<Platform> <method> requires account — MC #<id>"
   )
   ```
2. `serialize()` is EXEMPT from the NOT_IMPLEMENTED requirement — it SHOULD be operational even in STUB lifecycle because it needs no credentials (offline contract).
3. Once an implementation moves to SANDBOX_VERIFIED, no method may throw NOT_IMPLEMENTED for the sandbox environment. If a method is genuinely deferred (e.g., `parseIncoming()` for HR v1), the lifecycle state must remain STUB until all 4 methods are operational. Exception: `parseIncoming()` for HR is formally deferred to 90 days post-GA per Plan v3 §4d. The HR adapter will hold a partial SANDBOX_VERIFIED state tracked by the `AdapterConfig` feature flag with `reason = "parseIncoming deferred — Phase 1H.6"`.

Caller rules:

1. Before calling `submit()` or `pollStatus()`, callers MUST check:
   ```kotlin
   val config = adapterConfigRepo.find(jurisdiction, "EINVOICE")
       ?: throw AdapterException(NOT_IMPLEMENTED, ...)
   if (!config.enabled) throw AdapterException(NOT_IMPLEMENTED, ..., message="Adapter disabled: ${config.reason}")
   ```
2. `NOT_IMPLEMENTED` caught at the route handler level maps to HTTP 503 (Service Unavailable)
   with body `{"error": "ADAPTER_NOT_AVAILABLE", "market": "<jurisdiction>"}`, NOT HTTP 500.
   This is the stub plugin HTTP 500 risk mitigation from ADR-015 §5.3.
3. `serialize()` callers do NOT need to check AdapterConfig — serialize is always available.

Error code precedence when multiple codes could apply:

```
NOT_IMPLEMENTED > AUTH_INVALID_CREDENTIALS > VALIDATION_BUSINESS_RULE > NETWORK_TIMEOUT
```

If a STUB adapter is also missing credentials, `NOT_IMPLEMENTED` takes precedence. Lifecycle state check happens before credential check.

2.6 Secret Management — GCP Secret Manager Taxonomy

All adapter credentials follow the taxonomy defined in ADR-019 §2.5:

```
Bilko/{env}/{market}/{secret-name}
```

- `{env}`: `dev`, `stage`, `prod`
- `{market}`: `HR`, `RS`, `BA_FED`, `BA_RS`
- `{secret-name}`: platform-specific identifier (kebab-case)

HR Storecove secrets (provision after MC #8675):

| GCP Secret Manager path | Content | Access

binding |
| ---------- | ---------- | ---------- |
| `Bilko/stage/HR/storecove-api-key` | Storecove sandbox API key | Cloud Run SA `bilko-stage-sa` |
| `Bilko/prod/HR/storecove-api-key` | Storecove production API key | Cloud Run SA `bilko-prod-sa` |
| `Bilko/stage/HR/storecove-legal-entity-id` | Storecove legal entity ID (sandbox) | Cloud Run SA `bilko-stage-sa` |
| `Bilko/prod/HR/storecove-legal-entity-id` | Storecove legal entity ID (prod) | Cloud Run SA `bilko-prod-sa` |

Mounting in Cloud Run:

```yaml
# gcp-deploy.yml (Cloud Run --set-secrets pattern):
--set-secrets="STORECOVE_API_KEY=Bilko/stage/HR/storecove-api-key:latest,\
STORECOVE_LEGAL_ENTITY_ID=Bilko/stage/HR/storecove-legal-entity-id:latest"
```

Env var naming convention: `<PLATFORM>_<FIELD>`, uppercase, underscores.
Accessed in `StorecoveApiClient` via `System.getenv("STORECOVE_API_KEY")`.

Secret rotation policy:

- Rotate API keys every 90 days OR on any Storecove security notice, whichever comes first.
- Previous version retained in Secret Manager for 24h to allow graceful failover.
- Rotation event: create new secret version ? update Cloud Run env ? verify health endpoint ? delete previous version 24h later.

Never in source code or logs: API keys, legal entity IDs, OIB values, IBAN values. `StorecoveHrFiskEInvoiceAdapter.sanitizeForLog()` must be called on all Storecove response bodies before logging.

RS future secrets (Phase 1S):

GCP Secret Manager path	Content
`Bilko/stage/RS/sef-api-key`	SEF sandbox access token
`Bilko/prod/RS/sef-api-key`	SEF production access token
`Bilko/stage/RS/sef-username`	SEF API username
`Bilko/prod/RS/sef-username`	SEF API username (prod)

SEF uses OAuth2 with client credentials. The token endpoint is `https://efaktura.mfin.gov.rs/`
(Serbian Ministry of Finance). Exact credentials shape to be confirmed at Phase 1S kickoff.

2.7 Per-Platform Field Mapping

How `CanonicalInvoice` fields map to platform-specific XML/JSON:

CanonicalInvoice field	HR (UBL 2.1 / Peppol)	RS (SEF XML)	BA-FED	BA-RS
`supplier.taxId`	`AccountingSupplierParty/.../CompanyID @schemeID="9934"` (OIB)	`/Invoice/Seller/TaxId` (PIB)	TBD	TBD
`buyer.taxId`	`AccountingCustomerParty/.../CompanyID @schemeID="9934"` (OIB)	`/Invoice/Buyer/TaxId` (PIB)	TBD	TBD
`invoiceNumber`	`cbc:ID`	`/Invoice/InvoiceNumber`	TBD	TBD
`issueDate`	`cbc:IssueDate` (ISO 8601)	`/Invoice/IssueDate`	TBD	TBD
`typeCode.untdidCode`	`cbc:InvoiceTypeCode` (380/381/384)	`/Invoice/InvoiceType`	TBD	TBD
`currencyCode`	`cbc:DocumentCurrencyCode` + `@currencyID` on all amounts	`/Invoice/Currency`	TBD	TBD
`taxBreakdowns[].taxRate`	`TaxSubtotal/TaxCategory/Percent`	`/Invoice/TaxTotal/TaxRate`	TBD	TBD
`taxBreakdowns[].taxCategory`	`TaxSubtotal/TaxCategory/ID` (S/Z/E/K per EN 16931 BT-118)	Serbian code set	TBD	TBD

`paymentMeans.paymentReference`	`PaymentMeans/PaymentID` (HR "Poziv na broj")	`/Invoice/PaymentReference`	TBD	TBD
`paymentMeans.iban`	`PayeeFinancialAccount/ID`	`/Invoice/BankAccount/IBAN`	TBD	TBD
`adapterMetadata`	`"hr.supplierOib"`, `"hr.buyerOib"`, `"hr.pozivNaBroj"` (inbound)	`"rs.sefId"`, `"rs.supplierPib"`	TBD	TBD

SEF XML note: SEF does not use UBL 2.1. It uses a Serbian-specific XML schema published by the Ministry of Finance. The SEF adapter maps `CanonicalInvoice` ? SEF schema directly; it does NOT go through UBL. `EInvoiceAdapter.serialize()` returns the platform's native format.

BA adapters (Phase 1B): CPF and UINO platforms have no published API specifications as of 2026-05-13. Phase 1B cannot begin until regulatory mandates define the technical specification (~2027 per plan v3 context).

2.8 HR Reference Implementation Design Decisions

`StorecoveHrFiskEInvoiceAdapter` is the reference implementation. Future adapters MUST replicate these patterns:

Design decision	Location in reference impl	Rule for future adapters
PII field redaction before logging	Lines 24–59 (`REDACT_PII_FIELDS`, `sanitizeForLog`)	REQUIRED — GDPR / audit rules
Offline serialization (no credentials)	Lines 567–571 (`serialize()`)	REQUIRED per §2.1 contract
Idempotency key (SHA-256 of id + invoiceNumber)	Lines 591–600 (stub comment — activate post-#8675)	REQUIRED if platform supports
Credential validation on startup flag	Lines 83–138 (`validateOnStartup`, `validate()`)	REQUIRED — default false for tests
Error code mapping to `AdapterException`	Lines 469–515 (`StorecoveErrorMapper`)	REQUIRED — NEVER propagate native
Structured metrics recording (`StorecoveMetrics`)	Lines 537–540	REQUIRED — Prometheus counters
Tax ID format validation in `serialize()`	Lines 748–774 (OIB check)	REQUIRED — early error, no network
`document_id` for deduplication	Lines 420–437 (`StorecovePayloadBuilder.wrap()`)	REQUIRED if platform supports 409

---

3. Adapter Lifecycle Governance

3.1 AdapterConfig Feature Flag

All adapter network paths (`submit`, `pollStatus`, `parseIncoming`) are gated by an `AdapterConfig` row in the database. Defined fully in ADR-019 §2.4; referenced here:

```sql
CREATE TABLE adapter_config (
    id           UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    market       VARCHAR(8) NOT NULL,   -- TaxJurisdiction enum value
    adapter_type VARCHAR(32) NOT NULL,  -- 'EINVOICE', 'BANK_STATEMENT', etc.
    enabled      BOOLEAN NOT NULL DEFAULT FALSE,
    reason       TEXT,                  -- Human-readable status note
    updated_at   TIMESTAMPTZ NOT NULL DEFAULT now(),
    UNIQUE (market, adapter_type)
);
```

Seed row for HR STUB state (Flyway V17):

```sql
INSERT INTO adapter_config (market, adapter_type, enabled, reason)
VALUES ('HR', 'EINVOICE', false, 'Storecove account pending — MC #8675');
```

Row is flipped to `enabled = true` by the operator (not by code) after SANDBOX_VERIFIED transition is confirmed by Proveo evidence.

3.2 Adapter Versioning

Each adapter exposes:

```kotlin
val adapterVersion: String  // e.g. "1.0.0"
```

The `CountryPlugin` implementation declares a minimum adapter version. Incompatibility detected at startup ? application fails fast with a clear error (not silent degradation).

---

4. Consequences

4.1 Positive

- **Offline serialization.** `serialize()` contract requires no network. Enables invoice PDF preview, offline testing, and regression test suites without live platform credentials.
- **Uniform error handling.** `AdapterException` is the only exception type crossing the adapter boundary. Callers implement one error handler, not four platform-specific ones.
- **Lifecycle visibility.** `lifecycleState` is first-class. Dashboards show "HR adapter: STUB" and alert when a market operates in degraded state.
- **Canonical model.** `CanonicalInvoice` enables cross-market reporting and analytics.
- **NOT_IMPLEMENTED ? HTTP 503.** Clients receive a clean "feature not available" response instead of an HTTP 500 stack trace when an adapter is in STUB state.

4.2 Negative

- **SEF XML schema maintenance.** RS's SEF format changes without semantic versioning guarantees. The adapter must track schema changes proactively.
- **BA adapters are TBD.** Phase 1B work cannot begin until regulations define the spec.
- **4 methods = all or nothing lifecycle.** If `parseIncoming()` is the last unfinished method, the adapter cannot advance to SANDBOX_VERIFIED. The HR partial-SANDBOX exception (§2.5 rule 3) is a pragmatic workaround; it should not become a pattern.

4.3 Risks

- **CanonicalInvoice field gap.** A platform-specific required field has no canonical counterpart. **Resolution:** `adapterMetadata: Map<String, String>` for platform-specific extras until they generalise to first-class fields.
- **Storecove CustomizationID ambiguity (D3).** Two candidate CustomizationIDs — PEPPOL_BIS3 and HR_CIUS. **Resolution:** Verify with Storecove support before MC #8675 sandbox activation. This is a HALT item. Wrong choice ? all HR invoices rejected.
- **Storecove routing.network field (HALT-4).** Existing code does not include `routing.network` field. Verify with Storecove sandbox whether this is required.
- **Secret rotation lag.** Expired API key ? all `submit()` calls throw `AUTH_INVALID_CREDENTIALS`. **Mitigation:** 90-day rotation schedule + cert-expiry-monitor (Task 4.3 in Plan v3).
- **OIB validation at serialize() vs submit().** Validates early (offline) but couples format and validation logic. Accepted trade-off: early errors are better than late ones.

---

5. References

Reference	Path	Lines
`EInvoiceAdapter` interface	`apps/api/src/main/kotlin/no/alai/bilko/einvoice/EInvoiceTypes.kt`	200–224
`CanonicalInvoice` definition	`apps/api/src/main/kotlin/no/alai/bilko/einvoice/EInvoiceTypes.kt`	141–156
`AdapterLifecycleState` enum	`apps/api/src/main/kotlin/no/alai/bilko/einvoice/EInvoiceTypes.kt`	22–26
`AdapterErrorCode` enum + `AdapterException`	`apps/api/src/main/kotlin/no/alai/bilko/adapter/AdapterTypes.kt`	1–41
HR reference impl — full file		

	`apps/api/src/main/kotlin/no/alai/bilko/country/hr/StorecoveHrFiskEInvoiceAdapter.kt`	1–777
`StorecoveMetrics` (Micrometer counters)	`apps/api/src/main/kotlin/no/alai/bilko/country/hr/StorecoveMetrics.kt`	1–73
`StorecoveApiClient` (credentials + base URL)	`StorecoveHrFiskEInvoiceAdapter.kt`	77–179
`StorecoveOibValidator` (ISO 7064 MOD 11,10)	`StorecoveHrFiskEInvoiceAdapter.kt`	194–225
`StorecoveErrorMapper` (HTTP ? AdapterErrorCode)	`StorecoveHrFiskEInvoiceAdapter.kt`	469–515
PII sanitize helper (`sanitizeForLog`)	`StorecoveHrFiskEInvoiceAdapter.kt`	24–59
`HrUblBuilder` (UBL 2.1 offline build)	`StorecoveHrFiskEInvoiceAdapter.kt`	241–387
`StorecovePayloadBuilder` (wrap JSON + dedup D2)	`StorecoveHrFiskEInvoiceAdapter.kt`	418–450
ADR-019 §2.4 (AdapterConfig table)	`docs/architecture/ADR-019-INTEGRATION-ADAPTER-REGISTRY.md`	§2.4
Plan v3 §4d HR critical path (sandbox verification)	`~/system/specs/bilko-multi-market-architecture-plan-v3-2026-05-11.md`	147–176
Plan v3 §4b ADR-016 requirement	`~/system/specs/bilko-multi-market-architecture-plan-v3-2026-05-11.md`	125–126
ADR-bilko-003 §Layer 2 (EInvoice serialization)	`~/system/specs/bilko-multi-market-architecture-plan/ADR-bilko-003-market-abstraction-layers.md`	103–117

---

6. Approval

Status: Accepted

Unblocks:

- Phase 1H Task 1H.2: `PluginHR.generateEInvoiceXml()` delegation to `StorecoveHrFiskEInvoiceAdapter`
- Phase 1H Task 1H.4: DI wiring — lifecycle state check before submit/pollStatus dispatch
- Phase 1H Task 1H.6: Storecove submit() activation (after MC #8675)
- ADR-019: Integration Adapter Registry — `AdapterConfig` table and secret taxonomy

Role	Sign	Date
Finverge — Markos Zachariadis	Signed	2026-05-13
Architecture Lead (Petter Graff)	Signed	2026-05-13
CEO (Alem Baši?)	Not required for contract ADR	—

---

7. Document History

Date	Author	Change
2026-05-11	Markos Zachariadis / Petter Graff	v1 — Phase 0' initial (MC #100362)
2026-05-13	Petter Graff	v2 — MC #100585: Full lifecycle state machine with explicit transition criteria; sandbox validation matrix (5 invoice types for HR-FISK Storecove); NOT_IMPLEMENTED transition rules; GCP Secret Manager taxonomy with HR+RS secret paths; HTTP 503 mapping for NOT_IMPLEMENTED; HALT items D3/D4 documented; StorecoveMetrics and StorecoveApiClient cited explicitly

Revision #3
Created 2026-05-14 10:09:18 UTC by John
Updated 2026-06-14 20:03:16 UTC by John