



Bundesverband für Energie- und
Wasserdatenmanagement e.V.

ERP-Implementierungsleitfaden

Billing Result API v1.0

Version 1.0.3

Oktober 2026 | Zielgruppe: ERP-Anbieter / Softwareentwickler

ERP-Implementierungsleitfaden — bved Billing Result API v1.0

Version: 1.0.3

Datum: 2026-10-01

Hinweis: Dieser Leitfaden wurde mit Hilfe von KI erzeugt.

Zielgruppe: ERP-Anbieter / Softwareentwickler (Wohnungswirtschaft)

Spezifikation: bved Billing Result API v1.0.3 (OpenAPI: [bved-billing-result-1-0.openapi.yaml](#))

Inhalt

1. Einführung und Kontext
 2. Voraussetzungen
 3. Authentifizierung
 4. API-Endpunkte
 5. Vollständige Feldreferenz
 6. ERP-Datenmapping
 7. Implementierungsmuster
 8. Vollständige JSON-Beispiele
 9. Fehlerbehandlung
 10. Logging-Empfehlungen
 11. Implementierungs-Checkliste
 12. Testempfehlungen
-

1. Einführung und Kontext

Was ist die Billing Result API?

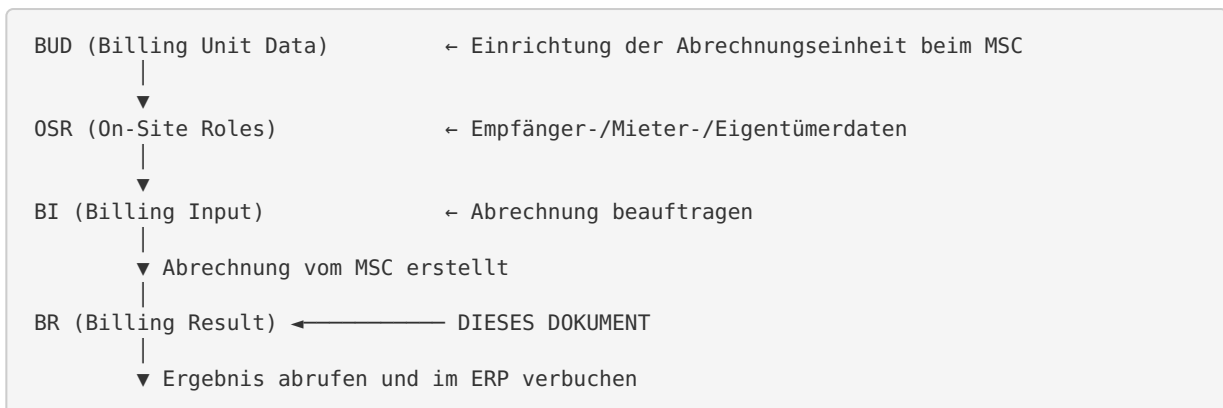
Die **bved Billing Result API v1.0** (BR) ist die REST-Schnittstelle, über die eine Hausverwaltung (Property Manager, PM) die **fertigen Ergebnisdaten einer Abrechnung** beim Messdienstleister (MSC) abrufen. Der MSC erstellt nach Abschluss der Abrechnung (die zuvor über die Billing Input API beauftragt wurde) die Ergebnisdaten; das ERP holt sie über die BR-API ab und verbucht sie.

Die BR-API ist eine **reine Lese-Schnittstelle** — sie besteht ausschließlich aus **GET**-Endpunkten. Es werden keine Daten gesendet, keine Transaktions-IDs erzeugt und kein Zustand verändert. Die Ergebnisdaten beziehen sich immer auf **einen Abrechnungszeitraum**.

Abgerufen werden:

- **Verfügbare Abrechnungszeiträume** je Abrechnungseinheit (max. drei Jahre zurück)
- **Verfügbare Abrechnungseinheiten** je Abrechnungszeitraum (mit Paginierung und inkrementellem Filter `updatedsince`)
- **Das Abrechnungsergebnis** selbst: Einzelergebnisse je Empfänger (Kostensalden, Verbräuche, Vorauszahlungen, CO₂-Kosten, §35a-Lohnkosten, Belegverweise) sowie das Gesamtergebnis der AE

Einordnung in die bved-API-Familie



Abhängigkeit: BR setzt voraus, dass für die Abrechnungseinheit und den Zeitraum bereits eine Abrechnung erstellt wurde (via BI beauftragt, vom MSC gerechnet). Ergebnisdaten erscheinen erst, nachdem der MSC die Abrechnung abgeschlossen hat.

Kernprinzip: Entdecken, abrufen, verbuchen

1. GET `.../billingperiods` → Welche Zeiträume liegen als Ergebnis vor?
 ODER
 GET `.../billingperiods/{ende}/billingunits` → Welche AE haben (neue) Ergebnisse für einen Zeitraum?
2. GET `.../billingperiods/{ende}/billingresult` → Das eigentliche Abrechnungsergebnis abrufen
3. Ergebnis im ERP verbuchen (Salden, Verbräuche, neue Vorauszahlungen, §35a, CO₂)
4. Belege (billingdocuments) über die Documents API herunterladen

Zwei zentrale Besonderheiten

1. **Aktualitätskennzeichen `updated`:** Jeder Zeitraum bzw. jede AE trägt einen `updated`-Zeitstempel (seit v1.0.3). In Verbindung mit dem Query-Parameter `updatedsince` lassen sich gezielt **neue oder nachträglich korrigierte** Abrechnungen abrufen – die Grundlage für einen inkrementellen Abgleich.
2. **Korrekturen überschreiben:** Wird eine Abrechnung vom MSC korrigiert, ändert sich ihr `updated`-Zeitstempel und das Ergebnis wird beim erneuten Abruf **vollständig neu** geliefert.

Das ERP sollte einen erneuten Import als **Ersetzung** des vorherigen Standes behandeln, nicht als Ergänzung.

2. Voraussetzungen

Voraussetzung	Details
OAuth 2.0 Credentials	Client ID und Client Secret vom MSC – nötig für Token-Abruf
<code>billingunitMscnumber</code>	9-stellige MSC-Nummer der AE (aus BUD-Setup) – für periodenbezogene Abfragen
Abrechnung erstellt	Ergebnisdaten liegen erst vor, nachdem die Abrechnung (via BI) beauftragt und vom MSC abgeschlossen wurde
Vertragliche Vereinbarung	Die Lieferung von Verbrauchs-/Ergebnisdaten setzt eine vertragliche Vereinbarung mit dem MSC voraus. Fehlt die Berechtigung, antwortet der MSC mit 403
HTTPS/TLS	Alle API-Aufrufe ausschließlich über verschlüsselte Verbindung
Basis-URL	Wird vom jeweiligen MSC bereitgestellt – kein gemeinsamer Endpunkt, jeder MSC hat seine eigene URL
Documents API (optional)	Zum Herunterladen der referenzierten Abrechnungsbelege (<code>billingdocuments</code>) wird die bved Documents API benötigt

3. Authentifizierung

Die API ist mit **OAuth 2.0 Client Credentials Flow** gesichert.

Token-Abruf

```
POST {tokenUrl}
Content-Type: application/x-www-form-urlencoded

grant_type=client_credentials
&client_id={client_id}
&client_secret={client_secret}
```

Die `tokenUrl` wird vom MSC mitgeteilt.

Token-Verwendung

```
GET /billingresult/v1/billingunits/123456789/billingperiods
Authorization: Bearer {access_token}
```

Implementierungshinweise

- Access Token vor Ablauf erneuern (TTL ist MSC-spezifisch — typisch 3600 Sekunden)
- Token serverseitig cachen, nicht per Request neu abrufen
- Bei 401 -Antwort: Token erneuern und Request einmalig wiederholen
- Credentials nicht im Quellcode oder in Logs speichern
- Der gleiche Token kann für alle bved-APIs desselben MSC wiederverwendet werden, sofern der MSC das so vorsieht

4. API-Endpunkte

Basis-URL: {msc-base-url}/billingresult/v1

Alle Antworten enthalten den Response-Header `X-API-VERSION` mit der vom MSC implementierten Version. Alle Endpunkte sind `GET` — die BR-API ist reine Lese-Schnittstelle.

4.1 GET — Verfügbare Abrechnungszeiträume einer AE

```
GET /billingunits/{billingunit}/billingperiods
```

Liefert die Liste der Abrechnungszeiträume, für die **Ergebnisdaten vorliegen** — begrenzt auf **maximal drei Jahre** rückwirkend.

Antwort bei Erfolg: 200 OK — Schema `BillingPeriods`:

```
{
  "periods": [
    { "from": "2023-01-01", "to": "2023-12-31", "updated": "2024-03-15T09:12:00Z" }
  ]
}
```

Verwendungszwecke: Ermitteln, für welche Zeiträume einer bekannten AE Ergebnisse abrufbar sind.

4.2 GET — Verfügbare Abrechnungseinheiten je Zeitraum

```
GET /billingperiods/{billing-period-to}/billingunits
GET /billingperiods/{billing-period-to}/billingunits?updatedsince={zeitstempel}&offset={n}
&limit={m}
```

Liefert — über **alle** dem Aufrufer zugänglichen AE hinweg — die Abrechnungseinheiten, für die im angegebenen Zeitraum Ergebnisse vorliegen. Paginiert.

Query-Parameter:

Parameter	Typ	Pflicht	Beschreibung
<code>updatedsince</code>	date-time	Nein	Nur AE zurückgeben, deren Abrechnung seit diesem Zeitpunkt neu oder aktualisiert wurde (Beispiel <code>2023-02-11T13:45:30Z</code>)
<code>offset</code>	integer	Nein	Anzahl zu überspringender Einträge (Default <code>0</code>)
<code>limit</code>	integer	Nein	Seitengröße; Vielfaches von 10, max. 100 , Default <code>20</code>

Antwort bei Erfolg: **200 OK** — Schema `BillingUnits` :

```
{
  "totalentries": 42,
  "billingunits": [
    { "billingunitMscnumber": "123456789", "billingunitPmnumber": "WEG-Muster-12",
      "updated": "2024-03-15T09:12:00Z" }
  ]
}
```

Verwendungszweck: Einstiegspunkt für das **Polling** — „welche AE haben seit meinem letzten Abruf (`updatedsince`) neue Ergebnisse für diesen Zeitraum?“

4.3 GET — Abrechnungsergebnis eines Zeitraums

GET `/billingunits/{billingunit}/billingperiods/{billing-period-to}/billingresult`

Liefert das vollständige Abrechnungsergebnis der AE für den durch sein **Enddatum** identifizierten Zeitraum: Einzelergebnisse je Empfänger und das Gesamtergebnis.

Antwort bei Erfolg: **200 OK** — Schema `Billing` (siehe §5.1 und Beispiel §8.3).

4.4 Path- und Query-Parameter

Parameter	Ort	Typ	Länge / Format	Beispiel
<code>billingunit</code>	Path	string	9 Stellen	<code>123456789</code>
<code>billing-period-to</code>	Path	date	YYYY-MM-DD	<code>2023-12-31</code>
<code>updatedsince</code>	Query	date-time	ISO 8601	<code>2023-02-11T13:45:30Z</code>
<code>offset / limit</code>	Query	integer	s. §4.2	<code>0 / 20</code>

Der Abrechnungszeitraum wird über sein Enddatum identifiziert (`billing-period-to`) — dasselbe Datum, das die Liste aus §4.1 / §4.2 als `to` liefert.

5. Vollständige Feldreferenz

Alle folgenden Strukturen sind **Antwort-Objekte** (die BR-API empfängt keine Request-Bodies).

5.1 Billing (Antwort von `GET .../billingresult`)

Feld	Typ	Pflicht	Beschreibung
<code>billingunitMscnumber</code>	string (9)	Ja	MSC-Nummer der Abrechnungseinheit
<code>billingunitPmnumber</code>	string (1–15)	Nein	ERP-eigene Nummer der AE
<code>billingdate</code>	date	Ja	Erstellungsdatum der Abrechnung
<code>billingperiod</code>	BillingPeriod	Ja	Abgerechneter Zeitraum (<code>from</code> , <code>to</code> , <code>updated</code>)
<code>individualbillingresults</code>	array	Ja	Einzelergebnisse je Empfänger (§5.2)
<code>totalbillingresult</code>	TotalBillingResult	Ja	Gesamtergebnis der AE (§5.7)

5.2 IndividualBillingResult (`individualbillingresults[]`)

Feld	Typ	Pflicht	Beschreibung
<code>residentialunitMscnumber</code>	string (4)	Ja	MSC-Nummer der Nutzeinheit
<code>residentialunitPmnumber</code>	string (1–20)	Nein	ERP-eigene Nummer der NE
<code>billingrecipient</code>	BillingRecipient	Ja	Empfänger der Abrechnung
<code>additionalbillingrecipient</code>	BillingRecipient	Nein	Zusätzlicher Empfänger (z.B. zweiter Mieter)
<code>usageperiod</code>	UsagePeriod	Ja	Nutzungszeitraum (<code>from</code> , <code>to</code> , <code>calendardays</code> , <code>degreedays</code>)
<code>billingbalances</code>	array	Ja	Kostensalden je Kostenblock (§5.3)
<code>newbudgetpayments</code>	array	Nein	Empfohlene neue Vorauszahlungen (BudgetPayment, §5.9)
<code>consumptions</code>	array	Nein	Verbräuche des Empfängers (Consumption, §5.9)

Feld	Typ	Pflicht	Beschreibung
wagecosts	array	Nein	§35a-EStG-Lohnkostenpositionen (CostPosition, §5.5)
co2costs	Co2Costs	Nein	CO ₂ -Kostenaufteilung des Empfängers (§5.4)
billingdocuments	array	Nein	Verweise auf Abrechnungsbelege (§5.6)

billingrecipient / additionalbillingrecipient — BillingRecipient:

Feld	Typ	Pflicht	Beschreibung
partnerPmnumber	string (1–20)	Ja	Referenz auf den Partner (wie via OSR gemeldet)
name	string	Nein	Nachname / Firmenname / Leerstandsbezeichnung
firstname	string (1–35)	Nein	Vorname (nur natürliche Personen)

5.3 BillingBalance (billingbalances[])

Kostensaldo je Kostenblock. Der Saldo ergibt sich als `totalcosts – budgetpayments`.

Feld	Typ	Pflicht	Beschreibung
costblock	enum	Ja	Kostenblock (siehe §5.9)
totalcosts	{ amounts }	Ja	Gesamtkosten des Blocks
budgetpayments	BudgetPayment	Nein	Geleistete Vorauszahlungen des Blocks
riskallocation	{ amounts }	Nein	Unterjähriger Abrechnungswert (UAW)
balances	{ amounts }	Ja	Saldo (negativ = Guthaben zugunsten Empfänger, positiv = Nachzahlung)

Jedes der Felder `totalcosts`, `riskallocation`, `balances` ist ein Objekt mit einem inneren `amounts`-Feld (Amount, §5.9).

5.4 Co2Costs (co2costs)

Kostenaufteilung gemäß CO₂KostAufG (Aufteilung zwischen Mieter und Vermieter).

Feld	Typ	Pflicht	Beschreibung
allocatedcosts	{ amounts }	Ja	Umlagefähige (aufgeteilte) CO ₂ -Kosten

Feld	Typ	Pflicht	Beschreibung
unallocatedcosts	{ amounts }	Ja	Nicht umlagefähige CO ₂ -Kosten
tenantshare	{ amounts }	Nein	Mieteranteil an den CO ₂ -Kosten
tenantpercentage	number (0–100)	Nein	Mieteranteil in Prozent

5.5 CostPosition (wagecosts[] — §35a EStG)

Lohnkostenpositionen für die steuerliche Absetzbarkeit haushaltsnaher Dienst- und Handwerkerleistungen (§35a EStG).

Feld	Typ	Pflicht	Beschreibung
costkey	string	Ja	bved-Kostenschlüssel (Muster <code>^[1-9]\d?\.\d{1,3}?\.\d{1,3}\$</code>)
costtext	string (1–25)	Ja	Textuelle Beschreibung der Kostenart
date	date	Nein	Rechnungsdatum
amounts	Amount	Ja	Beträge der Position
totalwageamount	WageShare	Ja	Lohnanteil auf AE-/Gesamtebene
individualwageamount	WageShare	Ja	Auf den Empfänger entfallender Lohnanteil

totalwageamount / **individualwageamount** — **WageShare**: **amount** (0–999999), **wagetype** (**PAR35A_ABS1** / **PAR35A_ABS2** / **PAR35A_ABS3**), **percentage** (0–100).

5.6 BillingDocument (billingdocuments[])

Verweis auf einen Abrechnungsbeleg. Der eigentliche Dateiinhalt wird über die **Documents API** anhand der **id** abgerufen.

Feld	Typ	Pflicht	Beschreibung
id	string	Ja	Dokument-ID für den Abruf über die Documents API
doctype	string	Ja	Dokumententyp (z.B. HKA-E Einzelabrechnung, HKA-G Gesamtabrechnung)
number	integer	Nein	Laufende Nummer
filename	string	Nein	Dateiname (z.B. abrechnung.pdf)

5.7 TotalBillingResult (`totalbillingresult`)

Gesamtergebnis der Abrechnungseinheit (aggregiert, nicht empfängerbezogen).

Feld	Typ	Pflicht	Beschreibung
<code>consumptions</code>	array	Nein	Gesamtverbräuche der AE (Consumption, §5.9)
<code>co2costs</code>	Co2Costs	Nein	Gesamt-CO ₂ -Kosten der AE (§5.4)
<code>billingdocuments</code>	array	Nein	Gesamtbelege (z.B. Liegenschaftsabrechnung, §5.6)

5.8 BillingPeriods / BillingUnits (Antworten der Listen-Endpunkte)

BillingPeriod (`periods[]` in §4.1; auch `billingperiod` in §5.1):

Feld	Typ	Pflicht	Beschreibung
<code>from</code>	date	Ja	Beginn des Abrechnungszeitraums
<code>to</code>	date	Ja	Ende des Abrechnungszeitraums (Identifikator)
<code>updated</code>	date-time	Nein	Letzte Aktualisierung der Ergebnisse dieses Zeitraums

BillingUnit (`billingunits[]` in §4.2):

Feld	Typ	Pflicht	Beschreibung
<code>billingunitMscnumber</code>	string (9)	Ja	MSC-Nummer der AE
<code>billingunitPmnumber</code>	string (1-15)	Nein	ERP-eigene Nummer der AE
<code>updated</code>	date-time	Nein	Letzte Aktualisierung der Ergebnisse dieser AE im Zeitraum

Die `BillingUnits` -Antwort trägt zusätzlich `totalentries` (integer, **Pflicht**) für die Paginierung.

5.9 Gemeinsame Bausteine

amounts — **Amount:** `grossamount`, `vatamount`, `netamount` (jeweils -9.999.999,99 ... 9.999.999,99), `vatrate` (0–100), `text`. Dezimaltrennzeichen ist der Punkt.

costblock — **CostBlock** (enum): `OVERALL`, `HEATING`, `HOT_WATER`, `ANCILLARY`, `HEATING_HOT_WATER`, `COLD_WATER`, `WASTE_WATER`, `COLD_WASTE_WATER`.

usageperiod — **UsagePeriod:** `from` (Ja), `to` (Ja), `calendardays` (1–366), `degreedays` (1–1000, Gradtagszahl).

consumptions[] — **Consumption:** `service` (`HEATING` / `HOT_WATER` / `COLD_WATER` / `WASTE_WATER`), `amount` (number), `unitofmeasure` (`M3` / `HCU` / `KWH` / `MWH` / `GJ`), `readingtype`

(`ESTIMATION` , `ESTIMATION_LASTYEAR` , `ESTIMATION_ETC` , `READING` , `READING_MAIN` , `READING_INTERMEDIATE`).

`budgetpayments` / `newbudgetpayments[]` – **BudgetPayment:** `costblock` (Ja), `amounts` (Ja), `validfrom` (date, ab wann gültig), `period` (integer 1–12; Zahlungen pro Jahr – 12 = monatlich).

6. ERP-Datenmapping

Anders als bei BUD/OSR/BI fließen die Daten hier **vom MSC ins ERP**. Das ERP liest das **Billing** - Objekt und verbucht die Bestandteile.

6.1 Kopf- und Zuordnungsdaten

API-Feld	ERP-Ziel	Hinweis
<code>billingunitMscnumber</code> / <code>billingunitPmnumber</code>	Abrechnungseinheit / Liegenschaft	Zuordnung über die ERP-eigene Nummer, wenn vorhanden
<code>billingdate</code>	Abrechnungsdatum	Buchungs-/Belegdatum
<code>billingperiod.from</code> / <code>.to</code>	Abrechnungszeitraum	<code>to</code> ist der Identifikator
<code>billingperiod.updated</code>	Import-Zeitstempel / Version	Für inkrementellen Abgleich merken (§7.4)

6.2 Einzelergebnis je Empfänger → Mieterkonto

API-Feld	ERP-Ziel	Besonderheit
<code>residentialunitMscnumber</code> / <code>residentialunitPmnumber</code>	Nutzeinheit	Zuordnung zur NE
<code>billingrecipient.partnerPmnumber</code>	Mieter-/ Empfängerkonto	Referenz aus OSR
<code>additionalbillingrecipient</code>	Zweiter Empfänger	Steht mit auf der Abrechnung
<code>usageperiod</code>	Nutzungszeitraum	Gradtagszahl (<code>degreedays</code>) ggf. für Plausibilisierung
<code>billingbalances[].balances.amounts.grossamount</code>	Nachzahlung / Guthaben	Negativ = Guthaben, positiv = Nachzahlung

API-Feld	ERP-Ziel	Besonderheit
<code>billingbalances[].totalcosts / .budgetpayments</code>	Kostennachweis je Block	Nachrichtlich zur Saldenherleitung
<code>newbudgetpayments[]</code>	Neue Vorauszahlungssoll	<code>validfrom</code> + <code>period</code> → Zahlplan; nicht automatisch aktivieren, sondern zur Prüfung vorlegen

6.3 Verbrauch, CO₂ und §35a

API-Feld	ERP-Ziel	Besonderheit
<code>consumptions[]</code>	Verbrauchshistorie	<code>service</code> + <code>unitofmeasure</code> + <code>readingtype</code> beachten (Ablesung vs. Schätzung)
<code>co2costs</code>	CO ₂ -Kostenaufteilung	Mieter-/Vermieteranteil (<code>tenantshare</code> / <code>tenantpercentage</code>)
<code>wagecosts[]</code>	§35a-Bescheinigung	<code>individualwageamount</code> je Empfänger für die Steuerbescheinigung; <code>costtext</code> als Positionstext

6.4 Belege → Documents API

API-Feld	ERP-Ziel	Besonderheit
<code>billingdocuments[].id</code>	Dokumentenablage	ID an die Documents API übergeben, um die Datei herunterzuladen
<code>billingdocuments[].doctype</code>	Dokumentklassifizierung	z.B. HKA-E (Einzelabrechnung) vs. HKA-G (Gesamtabrechnung)
<code>billingdocuments[].filename</code>	Dateiname	Für Ablage/Anzeige

Belege liegen nicht im BR-Ergebnis selbst. Das **Billing** -Objekt enthält nur **Verweise** (`id` , `doctype`). Den eigentlichen PDF-Inhalt liefert die bved Documents API.

7. Implementierungsmuster

7.1 Grundregel: Lesen ist gefahrlos wiederholbar

Alle BR-Aufrufe sind `GET` und **verändern keinen Zustand**. Sie können jederzeit gefahrlos wiederholt werden — Retry-Logik ist einfach (keine Idempotenz-Sonderfälle wie bei BI).

7.2 Muster 1: Neue Abrechnungen entdecken (Polling)

Der `updatedsince`-Filter macht den periodenbezogenen Listen-Endpunkt (§4.2) zum Polling-Einstieg:

```
Zustand im ERP: letzterAbruf = <Zeitstempel des letzten erfolgreichen Laufs>

GET /billingperiods/2023-12-31/billingunits?updatedsince={letzterAbruf}&offset=0&limit=100
  → Liste der AE, deren Abrechnung seit letzterAbruf neu/aktualisiert ist
  → für jede AE: Abrechnungsergebnis abrufen (Muster 2)

letzterAbruf = <jetzt>    // erst nach erfolgreichem Import aller Seiten setzen
```

7.3 Muster 2: Ergebnis abrufen und verbuchen

```
GET /billingunits/{bu}/billingperiods/2023-12-31/billingresult
  → Billing-Objekt

Pro individualbillingresults[:
  1. Empfänger + NE zuordnen (partnerPmnumber, residentialunitMscnumber)
  2. billingbalances → Salden je Kostenblock verbuchen
  3. balances.grossamount → Nachzahlung/Guthaben auf Mieterkonto
  4. newbudgetpayments → neuen Vorauszahlungsplan vorschlagen
  5. consumptions → Verbrauchshistorie fortschreiben
  6. wagecosts → §35a-Bescheinigung erzeugen
  7. billingdocuments → über Documents API herunterladen und ablegen
totalbillingresult → Gesamtwerte/Belege der Liegenschaft ablegen
```

7.4 Muster 3: Inkrementeller Abgleich über `updated`

Das ERP merkt sich je (AE, Zeitraum) den zuletzt importierten `updated`-Zeitstempel:

```
neuesUpdated = billingperiod.updated (bzw. billingunit.updated aus der Liste)

if (neuesUpdated > gespeichertesUpdated) {
  Ergebnis neu importieren    // Korrektur/Neuabrechnung
  vorherigen Import als ersetzt behandeln (überschreiben, nicht ergänzen)
  gespeichertesUpdated = neuesUpdated
} else {
  überspringen                // unverändert
}
```

7.5 Muster 4: Paginierung über Abrechnungseinheiten

Der Listen-Endpunkt §4.2 ist paginiert (`limit` Vielfaches von 10, max. 100). `totalentries` liegt in jeder Antwort vor:

```
offset = 0
do {
  GET /billingperiods/{ende}/billingunits?offset={offset}&limit=100[&updatedsince=...]
  verarbeite billingunits[]
  offset += 100
} while (offset < totalentries)
```

7.6 Muster 5: Belege herunterladen (Documents API)

Die `billingdocuments[].id` aus dem BR-Ergebnis wird an die Documents API übergeben. Empfehlung: Belege erst nach erfolgreichem Verbuchen des Ergebnisses laden und lokal referenzieren, damit ein Beleg-Downloadfehler den Import der Zahlen nicht blockiert.

7.7 Muster 6: Bekannte AE ohne Polling abrufen

Ist die AE bekannt und soll gezielt abgefragt werden (ohne periodenweiten Scan):

1. GET `/billingunits/{bu}/billingperiods` → verfügbare Zeiträume (max. 3 Jahre)
2. GET `/billingunits/{bu}/billingperiods/{ende}/billingresult`

8. Vollständige JSON-Beispiele

Alle Beispiele verwenden AE `123456789`, Zeitraum `2023-01-01 – 2023-12-31`.

8.1 GET .../billingperiods — Antwort

```
{
  "periods": [
    { "from": "2023-01-01", "to": "2023-12-31", "updated": "2024-03-15T09:12:00Z" },
    { "from": "2022-01-01", "to": "2022-12-31", "updated": "2023-03-20T11:05:00Z" }
  ]
}
```

8.2 GET .../billingperiods/2023-12-31/billingunits — Antwort (paginiert)

```
{
  "totalentries": 42,
  "billingunits": [
    { "billingunitMscnumber": "123456789", "billingunitPmnumber": "WEG-Muster-12",
      "updated": "2024-03-15T09:12:00Z" },
    { "billingunitMscnumber": "987654321", "billingunitPmnumber": "MFH-Beispiel-3",
      "updated": "2024-03-14T16:40:00Z" }
  ]
}
```

8.3 GET .../billingperiods/2023-12-31/billingresult – Antwort

```

{
  "billingunitMscnumber": "123456789",
  "billingunitPmnumber": "WEG-Muster-12",
  "billingdate": "2024-03-15",
  "billingperiod": {
    "from": "2023-01-01",
    "to": "2023-12-31",
    "updated": "2024-03-15T09:12:00Z"
  },
  "individualbillingresults": [
    {
      "residentialunitMscnumber": "0010",
      "residentialunitPmnumber": "EGL-Links",
      "billingrecipient": {
        "partnerPmnumber": "P-4712",
        "name": "Weber",
        "firstname": "Anna"
      },
      "usageperiod": {
        "from": "2023-01-01",
        "to": "2023-12-31",
        "calendardays": 365,
        "degreedays": 2894
      },
      "billingbalances": [
        {
          "costblock": "HEATING",
          "totalcosts": { "amounts": { "grossamount": 986.82 } },
          "budgetpayments": { "costblock": "HEATING", "amounts": { "grossamount": 1000.00 } },
          "balances": { "amounts": { "grossamount": -13.18 } }
        },
        {
          "costblock": "ANCILLARY",
          "totalcosts": { "amounts": { "grossamount": 412.50 } },
          "budgetpayments": { "costblock": "ANCILLARY", "amounts": { "grossamount": 360.00 } },
          "balances": { "amounts": { "grossamount": 52.50 } }
        }
      ],
      "newbudgetpayments": [
        {
          "costblock": "OVERALL",
          "amounts": { "grossamount": 120.00 },
          "validfrom": "2024-07-01",
          "period": 12
        }
      ],
      "consumptions": [
        { "service": "HEATING", "amount": 8200.0, "unitofmeasure": "KWH", "readingtype": "READING" },
        { "service": "HOT_WATER", "amount": 32.0, "unitofmeasure": "M3", "readingtype": "READING" }
      ],
      "wagecosts": [
        {
          "costkey": "2.10.15",
          "costtext": "Schornsteinfeger",
          "date": "2023-11-30",
          "amounts": { "grossamount": 119.00, "vatamount": 19.00, "netamount": 100.00, "vatrate": 19.0 },
          "totalwageamount": { "amount": 80.0, "wagetype": "PAR35A_ABS1", "percentage": 4.8 },

```

```

      "individualwageamount": { "amount": 12.0, "wagetype": "PAR35A_ABS1",
        "percentage": 4.8 }
    },
    "co2costs": {
      "allocatedcosts": { "amounts": { "grossamount": 145.22 } },
      "unallocatedcosts": { "amounts": { "grossamount": 245.11 } },
      "tenantshare": { "amounts": { "grossamount": 145.22 } },
      "tenantpercentage": 60.0
    },
    "billingdocuments": [
      { "number": 1, "doctype": "HKA-E", "filename": "Weber-2023.pdf", "id": "abcd-efgh-12345" }
    ]
  },
  "totalbillingresult": {
    "consumptions": [
      { "service": "HEATING", "amount": 82000.0, "unitofmeasure": "KWH", "readingtype": "READING" }
    ],
    "co2costs": {
      "allocatedcosts": { "amounts": { "grossamount": 1452.20 } },
      "unallocatedcosts": { "amounts": { "grossamount": 2451.10 } },
      "tenantpercentage": 60.0
    },
    "billingdocuments": [
      { "number": 1, "doctype": "HKA-G", "filename": "Gesamtabrechnung-2023.pdf", "id": "zzzz-yyyy-99999" }
    ]
  }
}

```

8.4 Inkrementeller Abruf — nur aktualisierte AE

```

GET /billingresult/v1/billingperiods/2023-12-31/billingunits?
updatedsince=2024-03-15T00:00:00Z&offset=0&limit=100

```

Liefert nur die AE, deren Abrechnung seit dem 15.03.2024 00:00 UTC neu erstellt oder korrigiert wurde.

9. Fehlerbehandlung

9.1 HTTP-Statuscodes

Status	Bedeutung	ERP-Maßnahme
200	Daten geliefert	Ergebnis verarbeiten / verbuchen
400	Technischer Validierungsfehler der Anfrage (z.B. ungültiges Datumsformat, ungültiges <code>limit</code>)	Request-Parameter korrigieren; kein Auto-Retry

Status	Bedeutung	ERP-Maßnahme
401	Token ungültig/abgelaufen	Token neu abrufen, Request einmalig wiederholen
403	Keine Berechtigung (fehlende vertragliche Vereinbarung für Ergebnis-/Verbrauchsdaten)	Vertragliche Klärung mit MSC; kein Auto-Retry
404	AE und/oder Abrechnungszeitraum nicht gefunden	<code>billingunitMscnumber</code> / <code>billing-period-to</code> prüfen; kein Auto-Retry
500	Technischer Fehler beim MSC	Exponentiellen Backoff anwenden; MSC kontaktieren

Hinweis: Die BR-Fehlerantworten sind rein technisch und tragen — anders als bei BI/OSR — **kein strukturiertes `Validation` -/ `shortmessage` -Objekt**. Es gibt kein Feld für fachliche Positionsmeldungen und keinen `409` -Konflikt (nichts wird geschrieben).

9.2 Abgrenzung „keine Daten“ vs. Fehler

Situation	Verhalten	ERP-Maßnahme
Zeitraum noch nicht abgerechnet	Zeitraum erscheint nicht in <code>.../billingperiods</code>	Kein Fehler — später erneut abfragen (Polling)
Zeitraum älter als drei Jahre	Nicht mehr in der Liste enthalten	Kein Fehler — Archivierung im ERP sicherstellen
AE/Zeitraum-Kombination existiert nicht	404	Nummern/Datum prüfen
Keine Änderung seit <code>updatesince</code>	leere / kürzere <code>billingunits</code> - Liste	Kein Fehler — nichts zu importieren

9.3 Retry-Strategie

Netzwerkfehler / Timeout (kein HTTP-Status):

- GET gefahrlos identisch wiederholen (Lesezugriff, kein Zustand)
- Nach 3 Fehlversuchen: Retry-Queue, erneuter Versuch nach 5 Min

HTTP 500:

- Exponentieller Backoff: 1 Min, 5 Min, 15 Min, 1 Std → dann Eskalation

HTTP 400:

- KEIN Auto-Retry – Anfrageparameter prüfen und korrigieren

HTTP 401:

- Token erneuern, einmaliger Retry

HTTP 403 / 404:

- KEIN Auto-Retry – manuelle Klärung (Vertrag bzw. Nummern/Datum)

9.4 Häufige Fehler und ihre Ursachen

Symptom	Wahrscheinliche Ursache	Lösung
404 beim Ergebnisabruf	Abrechnung noch nicht abgeschlossen oder falsches <code>billing-period-to</code>	Zeitraum via <code>.../billingperiods</code> gegenprüfen
403 trotz gültigem Token	Keine vertragliche Vereinbarung für Ergebnis-/Verbrauchsdaten	MSC kontaktieren
400 an Listen-Endpoint	<code>limit</code> kein Vielfaches von 10 oder > 100; ungültiges <code>updatedsince</code>	Parameter korrigieren (max. 100, Vielfaches von 10; ISO-8601-Zeitstempel)
Ergebnis unvollständig / veraltet	Korrektur beim MSC → höherer <code>updated</code> -Wert	Über <code>updated</code> erkennen und neu importieren (überschreiben)
Salden doppelt verbucht	Erneuter Import ohne Ersetzung des Vorstands	Import als Ersetzung je (AE, Zeitraum, <code>updated</code>) behandeln
Beleg fehlt	Nur Verweis (<code>id</code>) im Ergebnis, Datei liegt in Documents API	Beleg über Documents API laden

10. Logging-Empfehlungen

Feld	Pflicht	Zweck
Zeitstempel (ISO 8601)	Ja	Reihenfolge / Polling-Läufe rekonstruieren
HTTP-Methode + Pfad	Ja	Endpoint (Perioden / AE-Liste / Ergebnis) unterscheiden

Feld	Pflicht	Zweck
<code>billingunitMscnumber</code>	Ja	Nachvollziehbarkeit, MSC-Support
<code>billing-period-to</code>	Ja	Zeitraum-Zuordnung
<code>updatesince</code> (falls genutzt)	Empfohlen	Inkrementellen Abgleich nachvollziehen
HTTP-Status	Ja	Fehleranalyse
<code>billingperiod.updated</code> / <code>billingunit.updated</code>	Ja	Versions-/Korrekturerkennung dokumentieren
Anzahl <code>individualbillingresults</code> importiert	Empfohlen	Vollständigkeitsprüfung
<code>offset</code> / <code>limit</code> / <code>totalentries</code>	Empfohlen	Paginierung nachvollziehen
Import-Ergebnis (verbucht / übersprungen)	Empfohlen	Idempotenz des Imports belegen

11. Implementierungs-Checkliste

Setup

- ☐ OAuth 2.0 Client Credentials vom MSC erhalten
- ☐ `tokenUrl` und Basis-URL des MSC konfiguriert
- ☐ Vertragliche Vereinbarung für Ergebnis-/Verbrauchsdaten mit MSC geklärt
- ☐ (Optional) Zugriff auf die Documents API für den Belegabruf eingerichtet

Abruf

- ☐ `GET .../billingunits/{bu}/billingperiods` implementiert (verfügbare Zeiträume, max. 3 Jahre)
- ☐ `GET .../billingperiods/{ende}/billingunits` implementiert (inkl. `updatesince`, `offset`, `limit`)
- ☐ Paginierung über `totalentries` implementiert (`limit` ≤ 100, Vielfaches von 10)
- ☐ `GET .../billingresult` implementiert

Verarbeitung / Mapping

- ☐ `individualbillingresults[]` je Empfänger/NE zugeordnet (`partnerPmnumber` , `residentialunitMscnumber`)
- ☐ `billingbalances[].balances` → Nachzahlung/Guthaben verbucht (Vorzeichen beachtet)
- ☐ `newbudgetpayments[]` → Vorauszahlungsvorschlag (nicht automatisch aktiviert)
- ☐ `consumptions[]` → Verbrauchshistorie (Ablesung vs. Schätzung unterschieden)
- ☐ `wagecosts[]` → §35a-Bescheinigung (`individualwageamount`)
- ☐ `co2costs` → Mieter-/Vermieteranteil verarbeitet
- ☐ `billingdocuments[].id` → Belege über Documents API geladen
- ☐ `totalbillingresult` → Gesamtwerte/Belege der Liegenschaft abgelegt
- ☐ Import je (AE, Zeitraum, `updated`) als **Ersetzung** behandelt (kein Doppel-Verbuchen)

Inkrementeller Abgleich

- ☐ `updated` -Zeitstempel je (AE, Zeitraum) gespeichert
- ☐ Polling mit `updatedsince` implementiert; `letzterAbruf` erst nach Erfolg fortgeschrieben
- ☐ Korrektur (`updated` gestiegen) → Neuimport mit Überschreibung

Fehlerbehandlung

- ☐ `200` → Ergebnis verarbeitet
- ☐ `400` → Anfrageparameter korrigiert (kein Auto-Retry)
- ☐ `401` → Token-Refresh + einmaliger Retry
- ☐ `403` / `404` → kein Auto-Retry, manuelle Klärung
- ☐ `500` → Exponentieller Backoff
- ☐ Netzwerkfehler → gefahrloser GET-Retry
- ☐ „Noch nicht abgerechnet“ (Zeitraum fehlt in Liste) ≠ Fehler behandelt

12. Testempfehlungen

Testumgebung

Für Integrationstests steht die **bved Testplattform** zur Verfügung. Sie bildet das Verhalten eines echten MSC nach und antwortet auf BR-Anfragen mit realistischen Mock-Daten.

- **Testplattform / Swagger UI:** <https://bved-api.de/testplatform/swagger-ui/index.html>

Empfohlene Testszenarien

Test	Erwartetes Ergebnis
GET .../billingunits/{bu}/billingperiods	200 mit Liste verfügbarer Zeiträume (from , to , updated)
GET .../billingperiods/{ende}/billingunits ohne Filter	200 mit totalentries + billingunits[]
GET .../billingperiods/{ende}/billingunits? updatedsince=...	200 nur mit seither aktualisierten AE
GET .../billingperiods/{ende}/billingunits? limit=100&offset=100	200 , zweite Seite gemäß totalentries
GET .../billingresult (abgerechneter Zeitraum)	200 mit vollständigem Billing -Objekt
GET .../billingresult mit unbekannter AE/Zeitraum	404
GET mit ungültigem limit (z.B. 15 oder 150)	400
GET mit ungültigem updatedsince -Format	400
Abruf ohne vertragliche Berechtigung	403
Token-Ablauf während Request	401 , nach Token-Refresh erfolgreich
Erneuter Abruf nach Korrektur (höherer updated)	200 , Ergebnis vollständig neu → Überschreiben im ERP
Wiederholter identischer GET	200 , identisches Ergebnis (Lesezugriff, kein Zustand)