By Wilmer Toro

superservicebros.com

ServicePower Claims Status Check (A/R Payments Bot) — Deep Dive

What this automation solves

Warranty A/R gets painful when you're chasing dozens of claims across manufacturers and portals. The problem isn't "checking one claim." The problem is:

- a manufacturer sends an **A/R past due email** with a table of invoice/claim numbers
- you need to know **what ServicePower says** for each claim
- you need to log **payment status + payment reference** in one place
- and you need that to happen without someone doing manual portal lookups all day

This workflow automates exactly that: it parses the A/R email, extracts every line item, pushes them into Sheets, then queries ServicePower per claim and updates the same rows with status + payment data.

High-level architecture

This workflow has **two entry points** and **two "stages"**:

Entry points

1. **Gmail Trigger** (automatic) – runs when the A/R email arrives
2. **Manual Trigger** – lets you run it on demand (useful for testing)

Stage A — Parse the A/R email → write line items to Google Sheets

Stage B — For each "OUR INVOICE" → call ServicePower → update the row with claim status + payment info

---

Stage A — Email → Clean line items → Google Sheets

1) Gmail Trigger: catches the A/R email

The Gmail Trigger is configured to look for emails with the subject:

- `Concern Regarding Old A/Rs`

This is the "incoming signal" that your A/R report is ready to be processed.

---

2) "Extract Email" Code Node: cleans HTML and finds the table

This is a **heavy-lift** node and it's honestly the secret sauce.

It:

- strips HTML tags and normalizes whitespace
- extracts metadata like sender, subject, messageId
- detects the manufacturer name by pattern:
    - it looks for text between **"Claims Department"** and **"Dear Sir…"**
      SP Claims AR Payments Bot
- grabs summary stats from the email body:
    - number of items outstanding
    - total value
    - paid to date

    - days past due
- extracts ONLY the section under:
    - `LIST OF PAST DUE ITEMS`
    - then finds the header line:
      `YOUR P.O. ... OUR INVOICE ... BALANCE DUE`
    - then keeps everything after that as `tableSection`

This gives the AI a clean “table-like” block to parse.

---

3) Manufacturer mapping (critical for ServicePower lookup)

In the same “Extract Email” node, there’s a `manufacturerMap` mapping table that converts the email’s customer/manufacturer name into:

- `manufacturerName` expected by ServicePower
- `serviceCenterNumber` expected by ServicePower

Examples in your map:

- LGE → manufacturerName “LG”, service center “”
- GE → “GE_APPLIANCES”, “”
- Bosch/BSH → “BSH”, “”
- Electrolux/Frigidaire → “ELECTROLUX”, “”
- SquareTrade and Assurant mappings too

This is what makes the workflow multi-manufacturer instead of hardcoded to one.

---

4) AI Agent: extracts every past-due line item into JSON

The AI Agent is instructed to parse the “accounts receivable table data” and produce strict JSON with:

- `po_number` (YOUR P.O.) — can be null if blank
- `invoice_number` (OUR INVOICE)
- `billed_date` (converted to YYYY-MM-DD)
- `amount`
- `paid_to_date`

- `balance_due`

It also enforces:

- strip `$` and commas
- strict JSON only (no explanations)

Model used: `gpt-4.1-mini` with low temperature (0.1) for consistency.

---

5) “Code in JavaScript1” converts AI JSON into rows

This node:

- removes ```json fences if present
- parses the JSON
- loops each `line_item` and emits one n8n item per row
- normalizes empty PO numbers to `""` so Sheets won’t break on null
  SP Claims AR Payments Bot

---

6) Google Sheets: append or update

The workflow writes the parsed A/R list into your sheet:
“AR Claim Payments Bot” (Sheet1)

Mapped columns:

- OUR INVOICE
- YOUR P.O.
- BILLED DATE
- AMOUNT
- PdToDt
- BALANCE DUE

Matching column:

- **OUR INVOICE** (so the same invoice updates instead of duplicating)
  SP Claims AR Payments Bot

✅ At the end of Stage A, you have a clean A/R list centralized.

Stage B — Claim lookup in ServicePower → enrich the same sheet

7) HTTP Request to ServicePower “retrieval”

For each row, it calls ServicePower:

`https://claimworks.servicepower.com:8443/services/claim/v1/retrieval`

SP Claims AR Payments Bot

Request body includes:

- manufacturerName (from Extract Email mapping)
- serviceCenterNumber (from Extract Email mapping)
- claimNumber = the row’s “OUR INVOICE”
- authentication object
  SP Claims AR Payments Bot

Important note (security)

Right now, the ServicePower userId/password are hardcoded in the workflow JSON.

SP Claims AR Payments Bot

For the download version you give attendees, you should switch that to:

- n8n credentials, or
- environment variables, or
- encrypted variables in n8n

(So nobody pastes passwords into nodes.)

---

8) Google Sheets: update row with claim status + payment info

After the ServicePower response, the workflow updates the same row (matched by OUR INVOICE) with:

- SP STATUS = claimStatusDescription
- PAYMENT METHOD = paymentMethod
- PAYMENT REFERENCE = paymentTransactionNumber
- PAID LABOR = paidLaborAmount

This is the “money part” — now your A/R sheet isn’t just what you billed; it’s what ServicePower says happened.

What makes this workflow “operator-grade”

✅ It’s driven by the real artifact you already get (email)

You’re not asking staff to copy/paste claim lists or export reports.

✅ It normalizes manufacturer differences

That `manufacturerMap` is a scaling lever.

✅ It creates a single “source of truth” sheet

One list, continuously enriched.

✅ It reduces A/R follow-up time dramatically

Instead of checking claims one by one, you check **exceptions**.

How to configure it

If you want this to be a “download + plug credentials” setup, these are the only things they should touch:

1) Gmail Trigger

- change the Gmail account credential
- update the search query if their subject line differs
  (yours is `subject: Concern Regarding Old A/Rs`)

2) Manufacturer map

- update the `manufacturerMap` in “Extract Email” with:
  - the exact “Claims Department” name their emails contain
  - the matching ServicePower manufacturerName + serviceCenterNumber

3) OpenAI credential

- connect their OpenAI key (the AI Agent depends on it)

4) Google Sheet

- swap the spreadsheet ID + tab name
- ensure column headers match your mappings (“OUR INVOICE”, etc.)
  SP Claims AR Payments Bot

5) ServicePower credentials

- replace hardcoded authentication with their own secure credential method