Smart Router — deliver a spool file with its originating (READY / HELD) status preserved

Smart Router — deliver a spool file with its originating (READY / HELD) status preserved

Engine used in this walkthrough: TONSRENTER Screens: SR3 Last updated: 2026-08-28

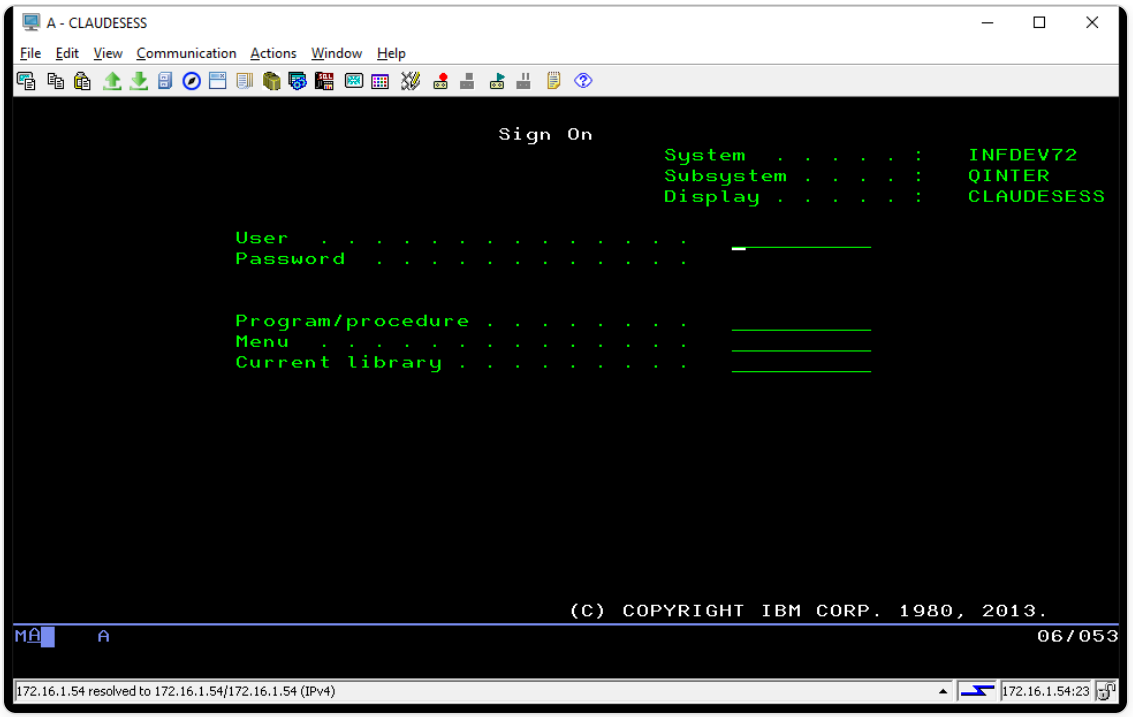
JIRA IDOCS-147

This guide walks through configuring and testing the **Keep Orig Status** feature: normally, when the Smart Router Engine routes a spool file to a target out queue, it delivers it in READY status regardless of how it originated. With this feature turned on for a target, a spool file that started HELD in the originating out queue is delivered still HELD.

Steps in this guide

1 Sign on

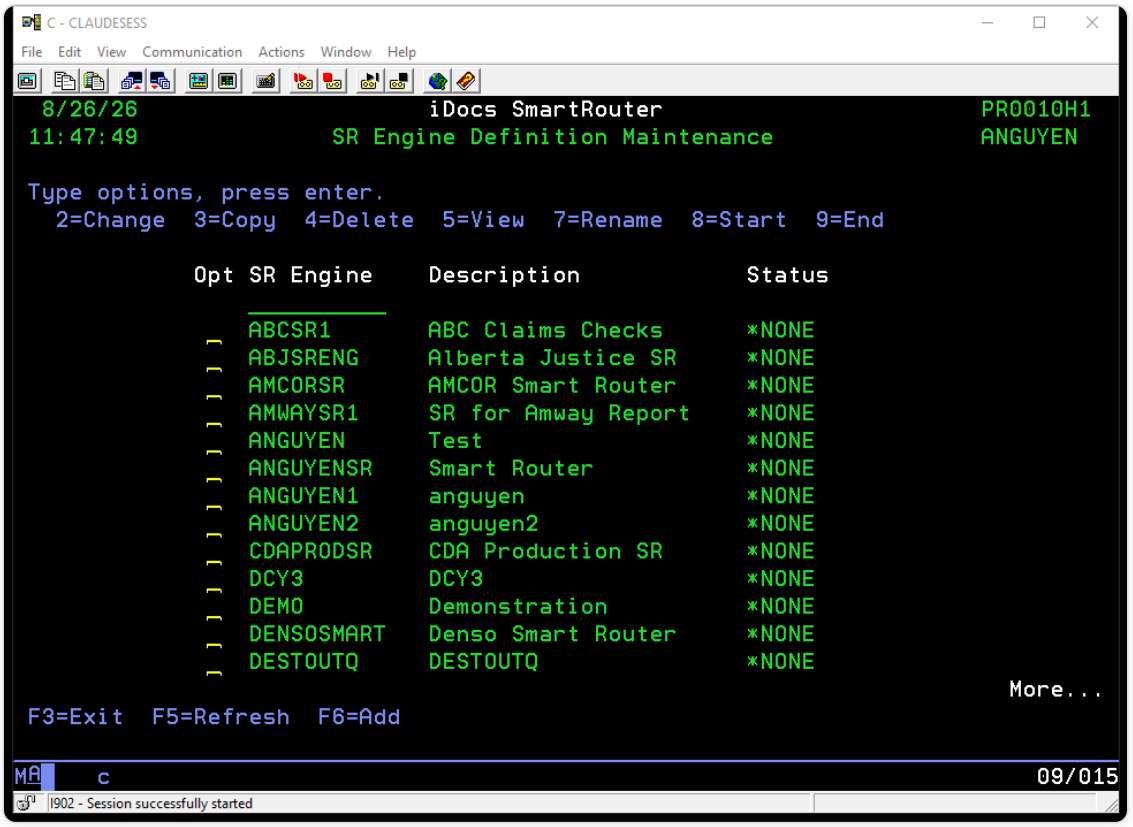
Sign on to the IBM i server as usual.



The standard Sign On screen.

2 Find your Smart Router engine

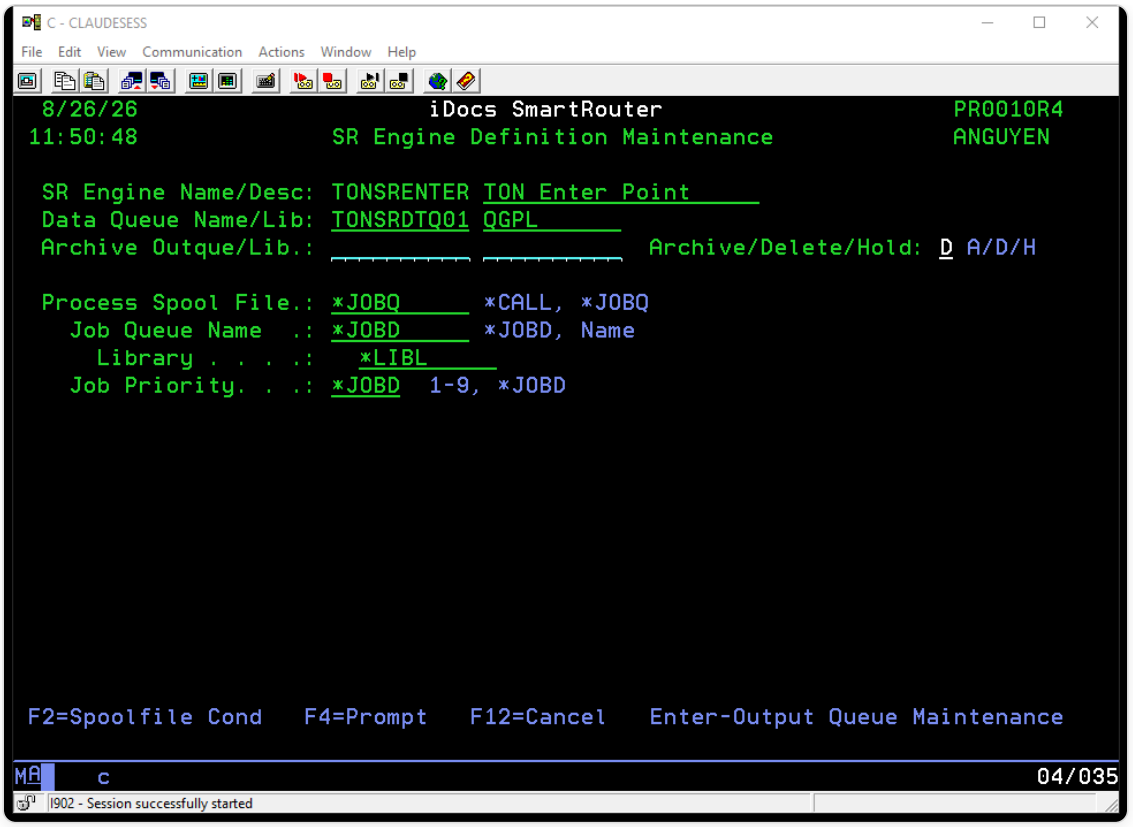
From a command line, run `SR3` to open **SR Engine Definition Maintenance** — the list of every configured Smart Router engine on the box. Page down (or use `F17=Position to`) to find the engine you want; this guide uses `TONSRENTER`.



SR3 — SR Engine Definition Maintenance, showing the full list of engines.

3 Review the engine's Data Queue / Archive settings

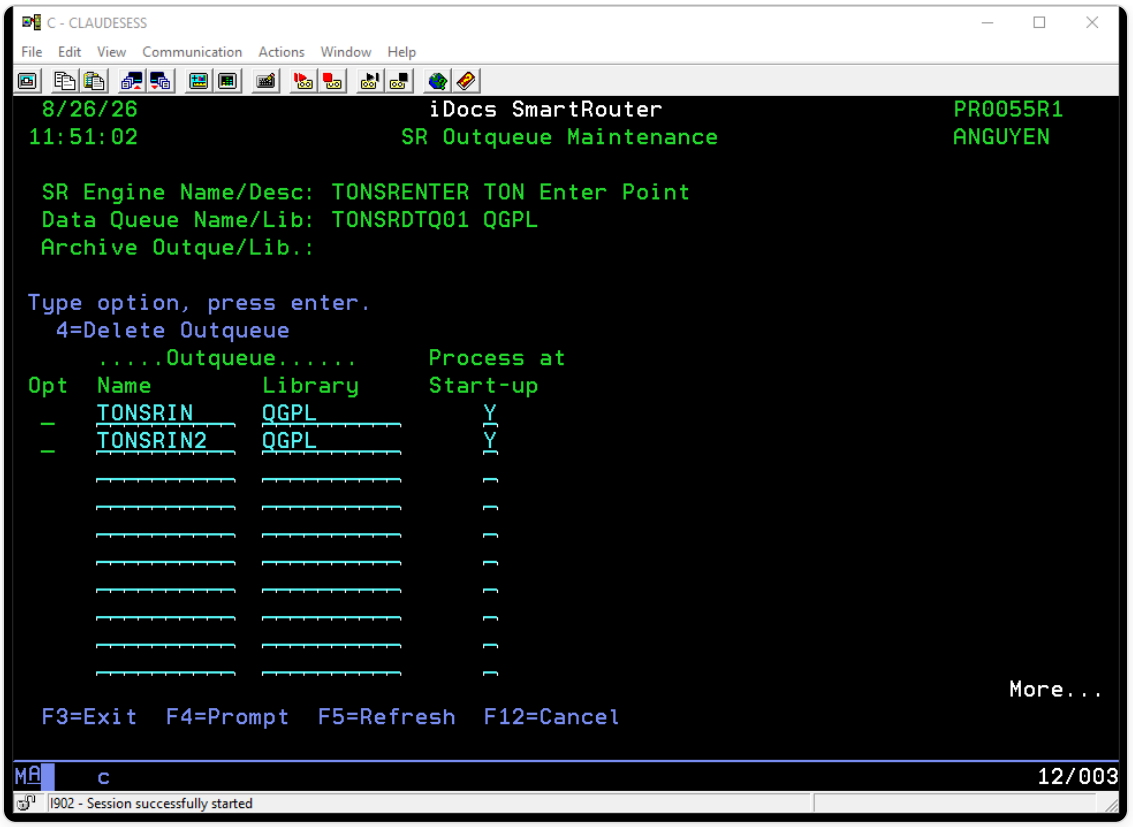
Type `2` next to the engine and press Enter. This shows the engine's data queue name, archive out queue, and spool-file processing job queue settings. Not something you need to change for this feature — shown here so you recognize the screen when navigating.



Engine-level settings reached via option 2=Change.

4 Review the monitored out queues

Pressing Enter again from the previous screen opens **SR Outqueue Maintenance** — the list of originating out queues this engine watches (`TONSRIN` and `TONSRIN2` for TONSRENTER), each with a **Process at Start-up** flag.

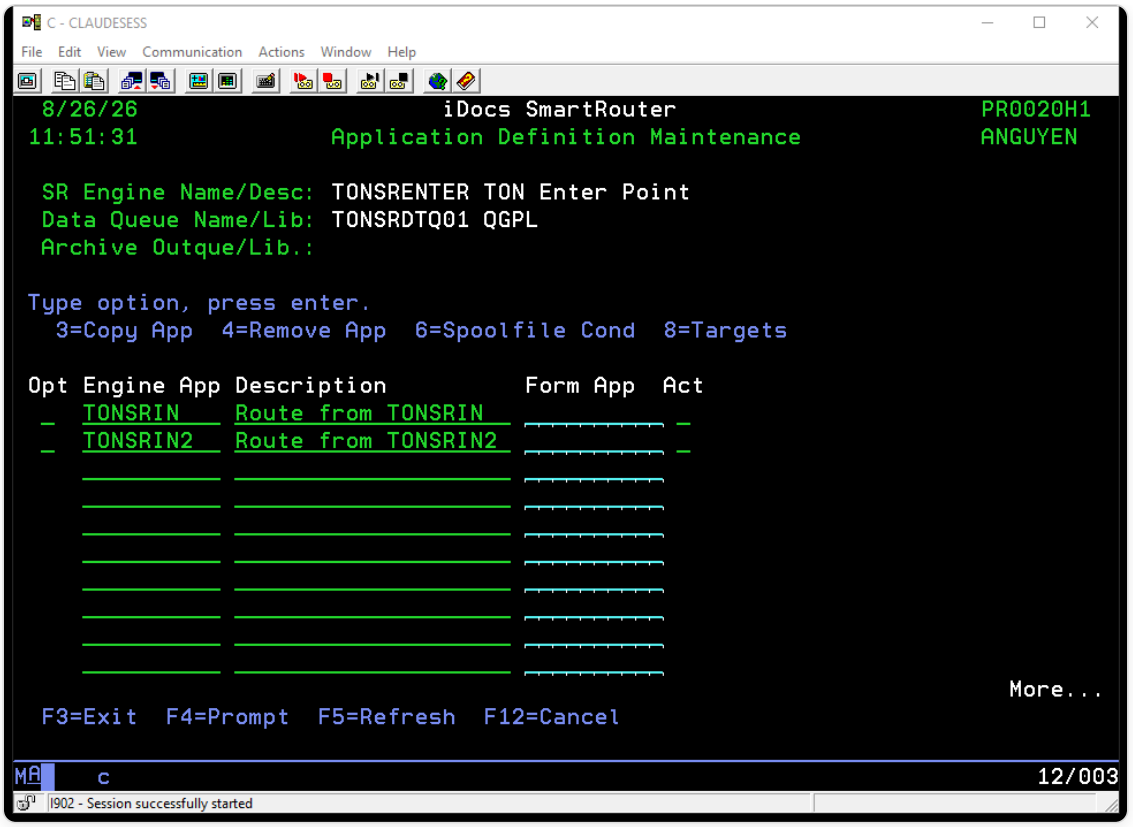


SR Outqueue Maintenance — the engine's monitored out queues.

> A second column, **Process HLD Live**, is now available on this same screen — see step 8 below for what it does and how to turn it on. It doesn't appear in the screenshot above since it wasn't enabled yet at the time.

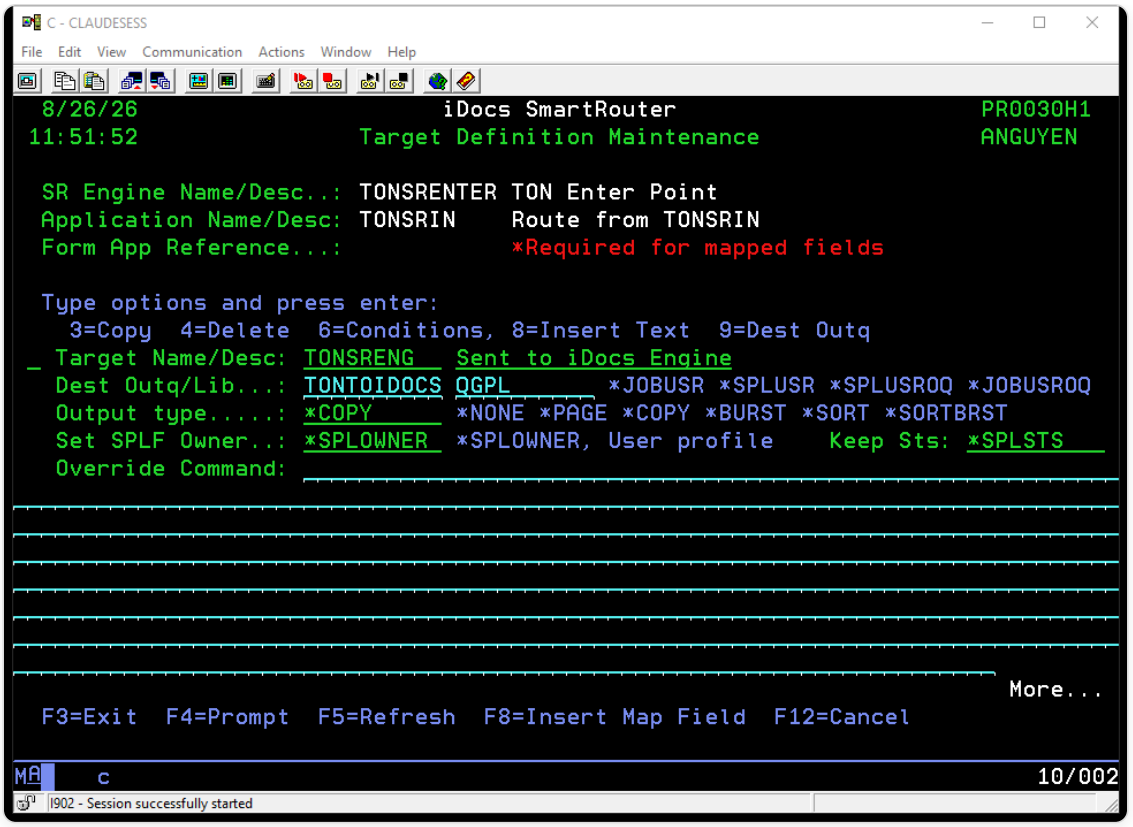
5 Turn on Keep Orig Status for a target

Applications and their targets aren't reached from the "Change" option above — go back to the engine list (`SR3`) and use **option `5` =View** instead. That opens **Application Definition Maintenance**, listing the applications (routes) configured for this engine.



Application Definition Maintenance — reached via SR3 option 5=View.

Type `8` next to an application to see its targets — this opens **Target Definition Maintenance**, where the `Keep Sts` field lives, on the far right of each target's settings.



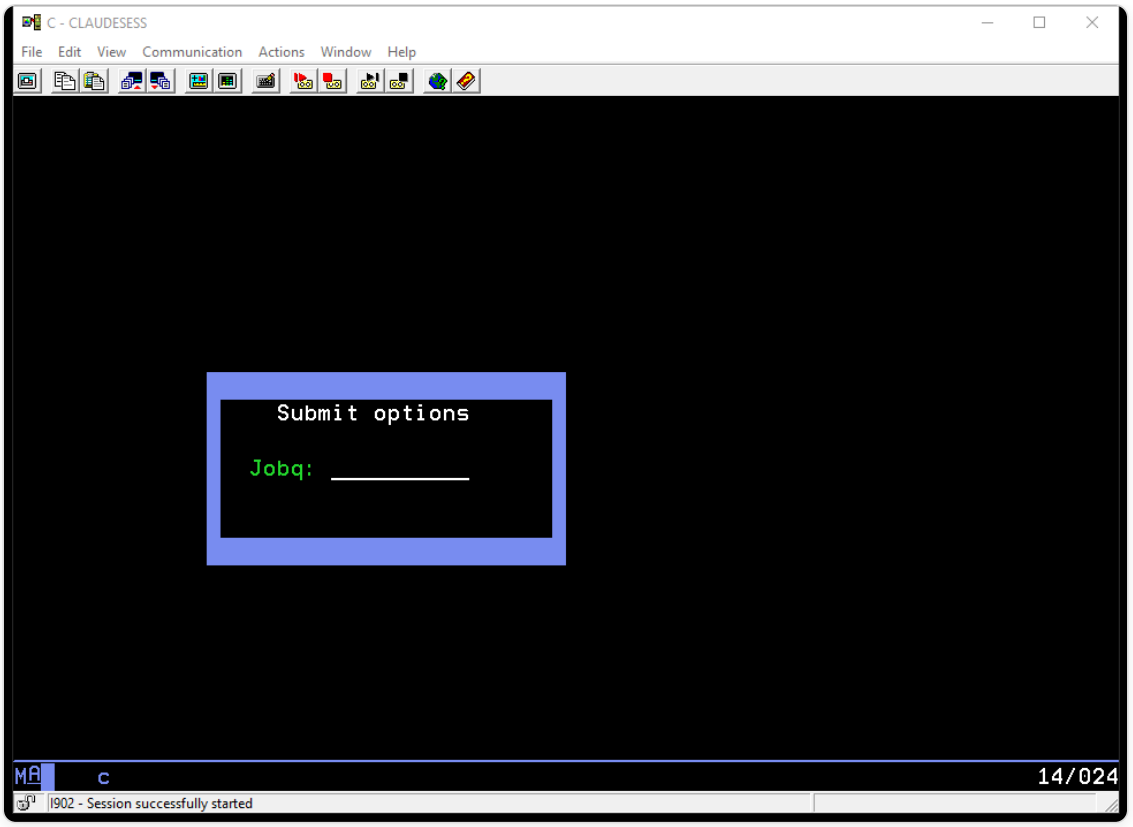
Target Definition Maintenance — `Keep Sts: *SPLSTS` is this feature turned on for the target.

Value	Behavior
`*SPLSTS`	Deliver the spool file to this target in the same status (READY or HELD) it originated with.
blank	Default/old behavior — always deliver as READY, regardless of origin.

> This field is directly editable in place — typing over it and pressing Enter changes the value immediately. Be careful not to accidentally type into the wrong field on this or the Application list screen above: doing so on the **App Name** field there renames the application (and everything under it) on the spot, with no confirmation prompt.

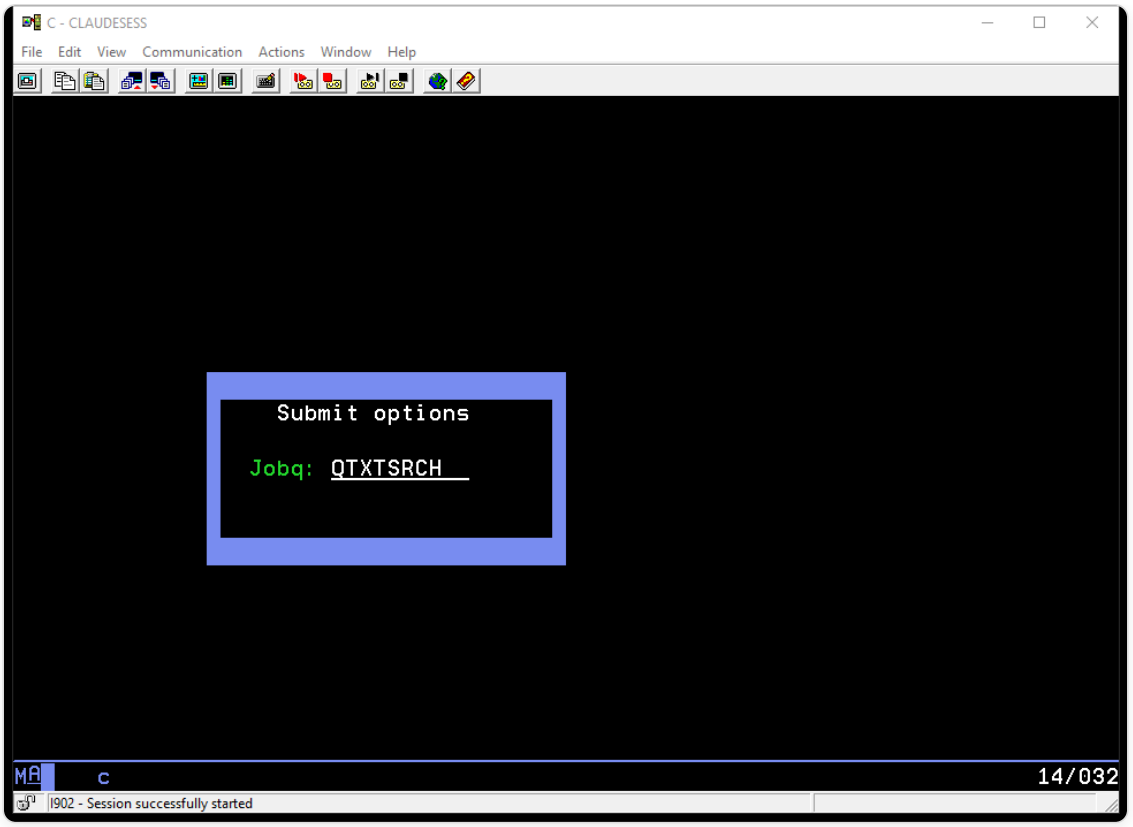
6 Start the engine

Back at the `SR3` engine list, type `8` next to the engine to start it. A "Submit options" pop-up asks for a job queue.



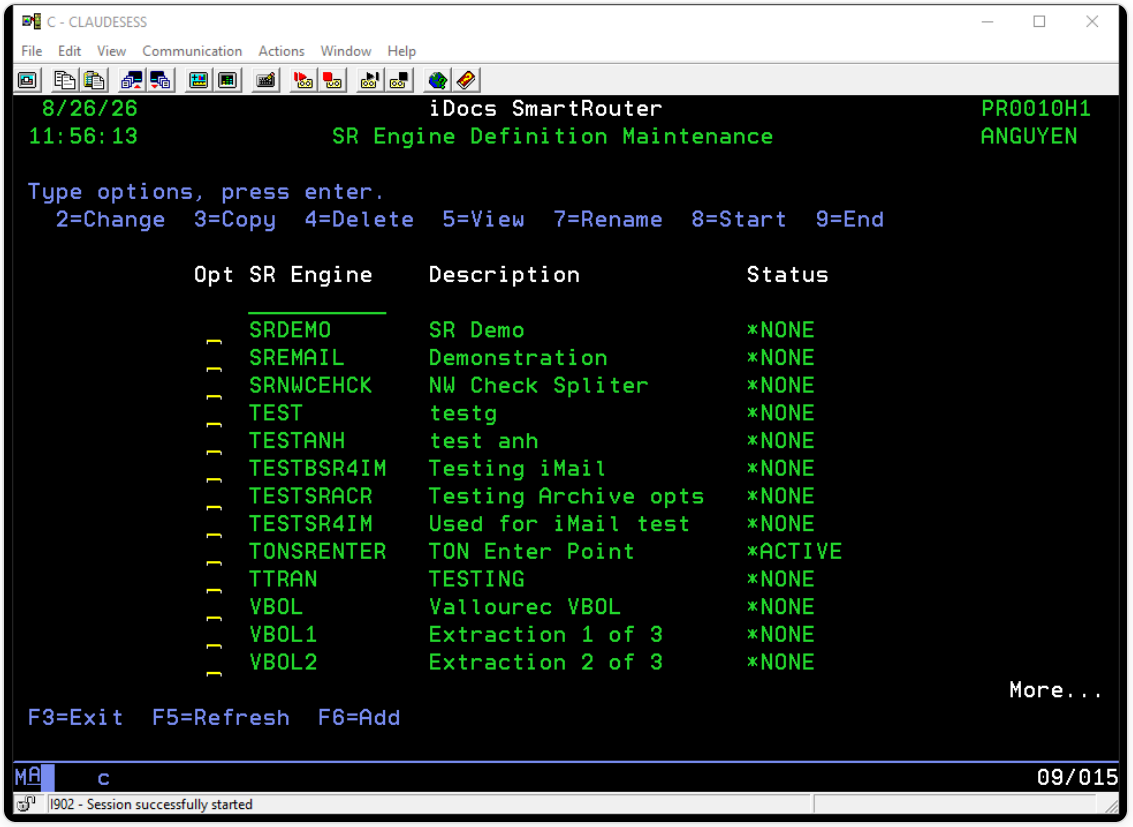
The Submit options pop-up shown by option 8=Start.

> Always type `QTXTSRCH` here — leaving it blank routes the job to `QBATCH`, where it sits stuck in `JOBQ` status and never actually starts.



Jobq filled in correctly before pressing Enter.

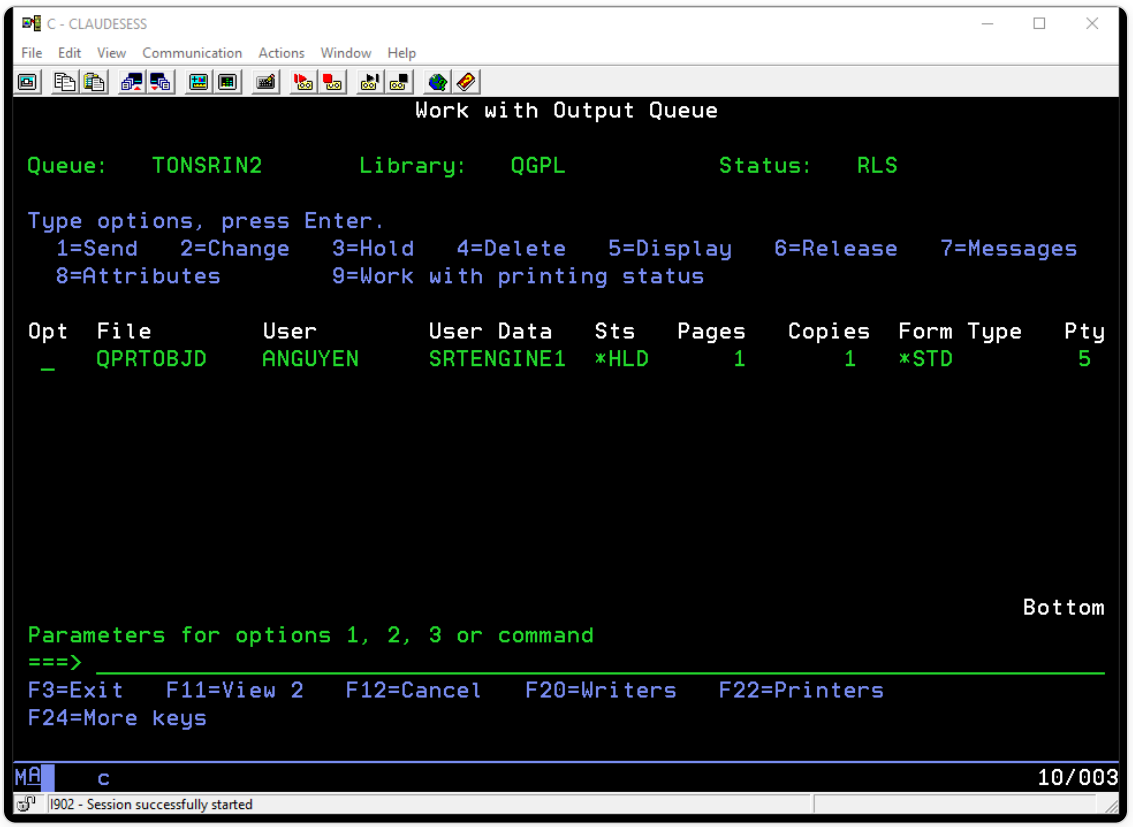
After pressing Enter, the engine list shows the engine's status as `*ACTIVE`:



TONSRENTER now running.

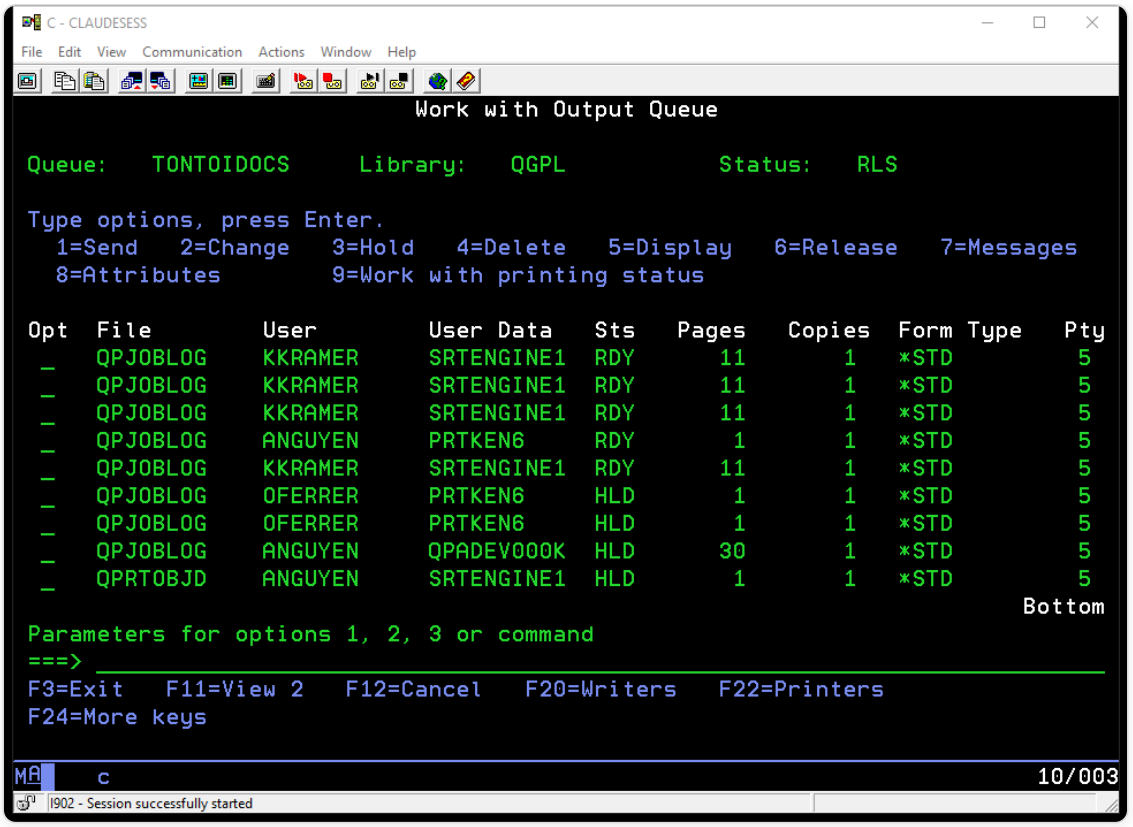
7 Test it — deliver a HELD spool file

Place (or find) a HELD spool file in one of the engine's originating out queues — here, `TONSRIN2`:



A HELD spool file waiting in the originating out queue TONSRIN2.

Once the engine picks it up and routes it, check the target out queue (`TONTOIDOCS`) — the delivered copy is still **HLD**, matching where it started:



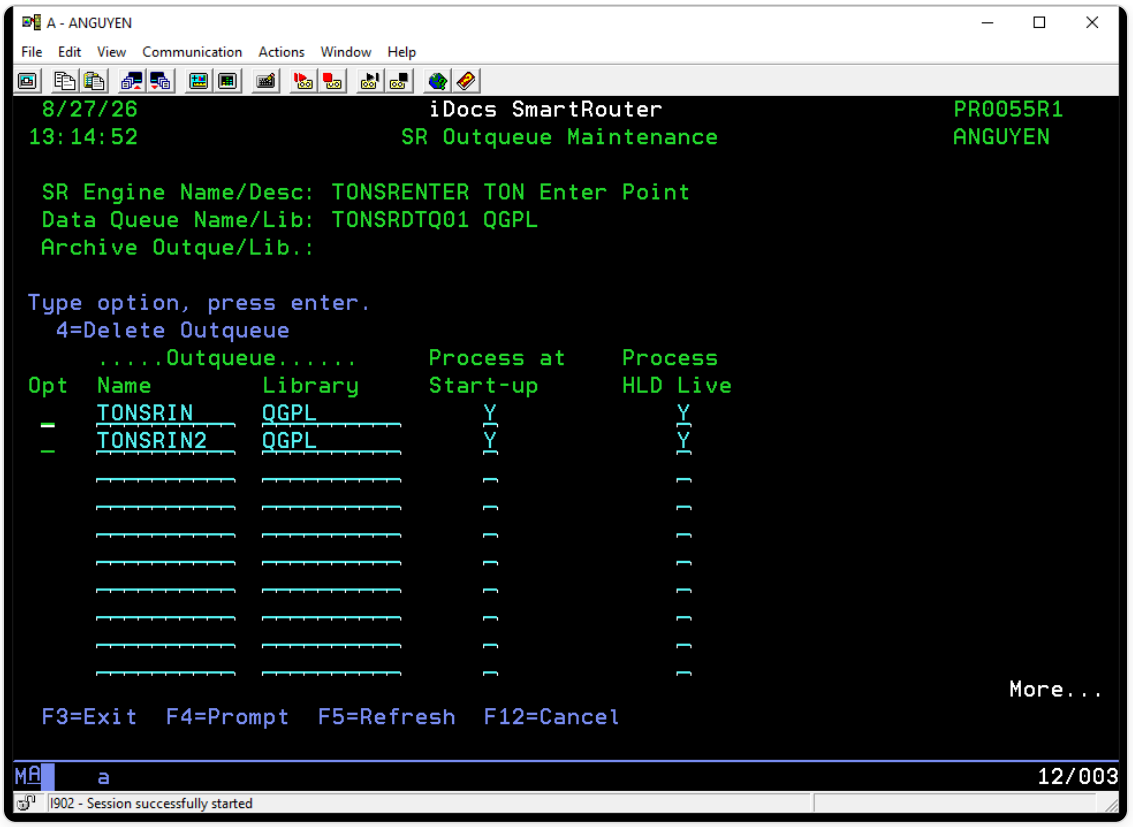
Delivered to the target out queue, status preserved as HLD.

> Confirmed working: the delivered spool file's status matches its origin, which is the entire point of this feature.

8 Process HLD Live — pick up HELD files without restarting

Background: the feature in steps 1–7 only controls the *delivered* status — it doesn't change whether the engine notices a spool file in the first place. By default, while an engine is running continuously (not just when it starts), it only reacts live to spool files that arrive in **READY** status. A file that arrives already HELD — the normal case for many originating applications — used to be picked up only if it happened to already be sitting in the queue at the exact moment the engine *started*. A HELD file arriving while the engine was already running and idle would sit there unnoticed until the next restart.

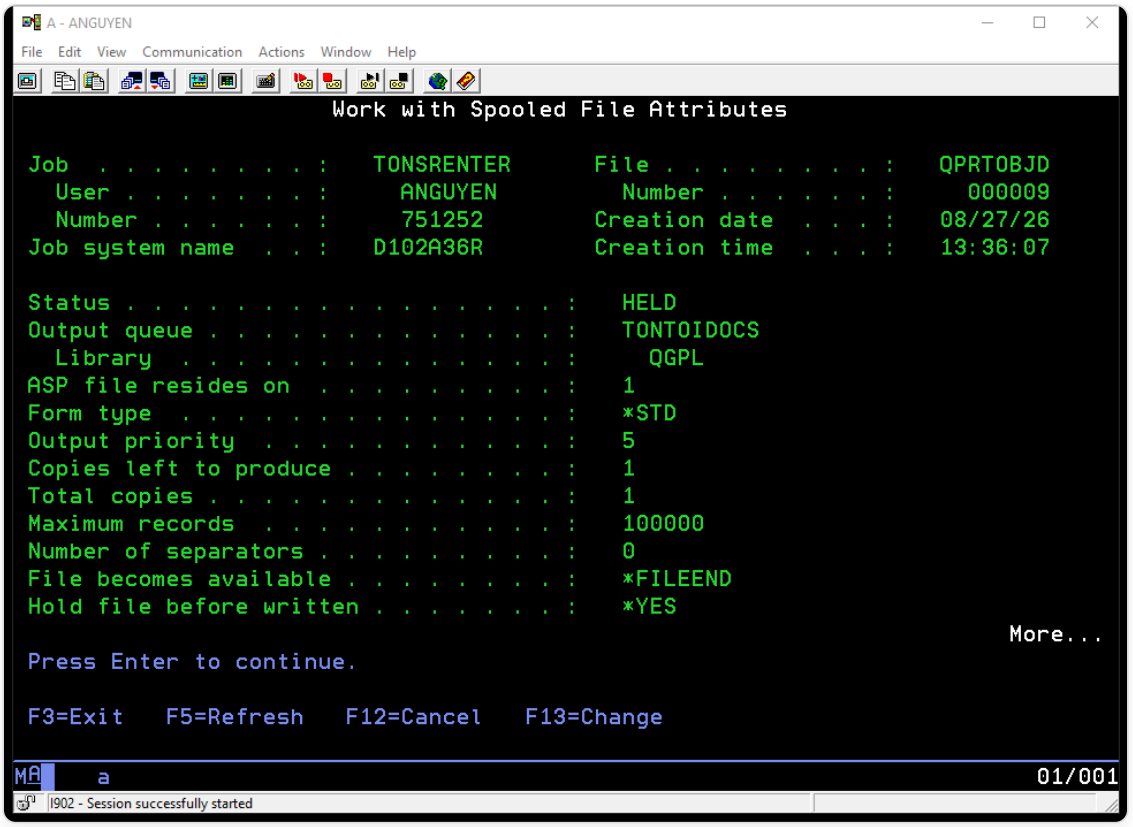
Process HLD Live is a new opt-in column on the SR Outqueue Maintenance screen (step 4 above) that fixes this. When turned on for at least one of an engine's monitored out queues, the engine periodically re-checks all of its out queues (every 30 seconds) for anything the live READY-only notification can't see — HELD files included — instead of waiting indefinitely between checks. Engines that don't turn it on keep running exactly as before; there's no behavior change unless you opt in.



Process HLD Live set to `Y` for both of TONSRENTER's out queues (TONSRIN and TONSRIN2).

> This setting is only read when the engine *starts* — after turning it on, end and restart the engine (step 6 above) for it to take effect.

Confirmed working live (2026-08-28): with the engine already running and idle — not just-started — a HELD spool file was placed directly into TONSRIN2. No restart. Within seconds it was picked up and delivered to TONTOIDOCS still showing **HELD** status, confirmed by the delivered spool file's own attributes:



Delivered spool file's attributes — Job `TONSRENTER`, Status `HELD`, Output queue `TONTOIDOCS` — proving the engine picked it up live, without a restart.

> This is exactly the scenario that used to fail every time before this fix. It now works.

See IDOCS-147.html in this same folder for the full technical write-up of both parts of this change, and the appendix below for the full implementation design behind Process HLD Live specifically (also available as raw markdown: idocs-147-live-hld-pickup-design.md).

A Appendix: technical design — live HLD pickup

Full implementation design for the Process HLD Live feature (step 8 above). Converted from `idocs-147-live-hld-pickup-design.md` in this same folder.

> **Status:** implemented (2026-08-21, tag `60825`), compiled into library `IDOCS6` — the library the live TONSRENTER engine actually runs from — and confirmed working live (2026-08-28): a HELD spool file placed into `TONSRIN2` while the engine was already running and idle was picked up and delivered to `TONTOIDOCS` still `HELD`, without a restart. See "Deploy steps" below for what that involved.

Scope: `PREPROC300/QRPGLESRC/PRR0459.RPGLE` (Smart Router dispatcher — shared by every SR engine on the box), plus its config file `PRDOUTQUE` and maintenance screen `PRD0055`.

Problem

IDOCS-147 (fixed in `PRR0100.RPGLE`) makes the router *deliver* a spool file with the same status it originated with. But that only matters if the router actually *picks up* the file in the first place. Confirmed by direct testing (2026-08-19/20) plus confirmation from Orlando (customer-facing): `PRR0459` only reacts live to `*READY` spool files. A spool file that arrives already `*HLD` — which per Orlando is the **normal, common case** for TONSRENTER's originating queues (the reporting app itself creates them held) — was only ever picked up if it happened to already be sitting in the queue at the moment the engine was *started*. Once the engine is running continuously (the normal production state), a newly arrived HELD file was invisible to it forever, until the engine happened to restart.

Why: three separate pickup paths in PRR0459, only one covers all statuses

Path	When it runs	Status filter	Where
`SendAllSplf()`	once, at engine startup, per outqueue with `PRDOUTQUE.OQHLDSPL='Y'`	`dsFilter.splStat = '*ALL'` — all statuses	PRR0459.RPGLE:761-771
DTAQ live notification	continuously, while the engine runs (TONSRENTER's actual mode — `PRDMASTER.DTAQ='TONSRDTQ01'`, not blank)	not filtered in RPG — the OS-level `CHGOUTQ...DTAQ()` feature itself only ever fires a `*SPOOL` entry when a spooled file becomes RDY/eligible for a writer. Empirically confirmed (HELD-on-creation files placed into a running engine never generated a DTAQ entry across 3 separate attempts), not something the RPG code controls.	PRR0459.RPGLE:555-597, `$dataq()` at 605-627
`$polling()` (alternative	continuously, on a `DLYJOB` interval	`ds_f_elem.elemdat = '*READY'` —	PRR0459.RPGLE:903-911 (literal at line

engine mode, used only when `PRDMASTER.DTAQ` is blank)		**hardcoded** in the RPG itself	908)

So: startup sweep = all statuses; both ongoing paths = RDY only. Nothing ever re-checks a monitored outqueue for HELD files once the engine has finished starting up.

Fix approach: opt-in periodic HELD sweep, added to the DTAQ-wait loop

Chosen because `PRR0459` is shared by 100+ configured SR engines on this box (per `SR3`'s listing) — the fix must be **zero-behavior-change for every engine that doesn't explicitly opt in**.

1. New config field — PRDOUTQUE.OQHLDLIVE (1A, Y/N, mirrors OQHLDSPL)

- `PREPROC300/QDDSSRC/PRDOUTQUE.PF` — add `OQHLDLIVE 1A TEXT('Process HLD live')` after the existing `OQHLDSPL` field (same pattern as the 13760-tagged `OQHLDSPL` addition).
- Defaults to blank/N for every existing row via `ALTER TABLE ... ADD COLUMN ... WITH DEFAULT` — matches how `OQHLDSPL` and IDOCS-147's `PRDTARGET.SPLSTS` were added.
- Set to `Y` only on the `TONSRENTER` / `TONSRIN` and `TONSRENTER` / `TONSRIN2` rows once deployed.

2. Screen — PRD0055.DSPF + PRR0055.SQLRPGLE

- DDS: add a new subfile column next to the existing `HLDSPL` field (PR0055S1 record, currently at row 12 col 35 — see PRD0055.DSPF:69). New field e.g. `HLDLIVE` 1A at col ~45, with a two-line header at rows 10-11 (mirroring "Process at" / "Start-up" at PRD0055.DSPF:49-50) reading something like "Process" / "HLD Live".
- RPG: `PRR0055.SQLRPGLE` handles `OQHLDSPL` in exactly two places worth mirroring for the new field:
  - **Load** (subfile populate, embedded SQL): PRR0055.SQLRPGLE:415,429 — SQL `SELECT ... OqHldSpl ... :HldSpl` — add `OqHldLive` to the select list and a matching host variable/screen field.
  - **Save** (native record I/O on `PrdOutQue`): PRR0055.SQLRPGLE:288-297 — `Chain (prenam:outq:outqlb) prdoutque; OqHldSpl = HldSpl;` — add `OqHldLive = HldLive;` alongside it. Also mirror the default-if-blank guard at PRR0055.SQLRPGLE:264-265 (`If HldSpl <> 'N' and HldSpl <> 'Y'; HldSpl = 'N'; EndIf;`) for the new field.

3. PRR0459.RPGLE — the actual dispatcher change

All changes confined to the `Else` branch at line 545 (the DTAQ-mode loop; polling mode is untouched since TONSRENTER doesn't use it, and this fix doesn't need to touch it).

a. During outqueue load (loop at PRR0459.RPGLE:507-530, where `outQueueDS` is built): read the new `o_oqhldlive` field from each `PRDOUTQUE` row (already in the record buffer once the DDS/PF change lands — no extra I/O needed) and set a single job-scope indicator, e.g. `bLiveHldEnabled`, to `*ON` if **any** row for this engine has `o_oqhldlive = 'Y'`. Simplification: this is engine-wide, not per-queue — if any one of an engine's monitored outqueues opts in, the periodic sweep checks *all* of that engine's outqueues on each timeout. Chosen over per-queue tracking because the sweep is cheap (it's the same `SendAllSplf()` call the startup path already trusts) and avoids extending the `outQueueDS` array shape.

b. Bound the wait instead of waiting forever — PRR0459.RPGLE:553 currently sets `@@wait = -1` (wait indefinitely) before the receive loop. Change to:

```
If bLiveHldEnabled;
  @@wait = 30;   // seconds; re-checks for HELD arrivals on timeout
Else;
  @@wait = -1;   // unchanged for every engine that hasn't opted in
EndIf;
```

30s is a starting point, not a hard requirement — trades detection latency for I/O volume (each tick calls `QGYOLSPL` once per monitored outqueue via `SendAllSplf`). Should be confirmed with Orlando/whoever owns acceptable latency for this use case before finalizing.

c. Detect a timeout vs. a real message, and sweep — `$dataq()` (PRR0459.RPGLE:607-627) calls `QRCVDTAQ` with `lenDtaqData` as an in/out parameter; per the API contract it comes back `0` if the wait elapses with nothing received (this is the standard, documented way to distinguish a timeout from an actual entry — not an assumption, it's how QRCVDTAQ always behaves). In the main loop (PRR0459.RPGLE:555-597), immediately after the `$dataq()` call at line 557:

```
$dataq();
eval dataqDS = dataQData;

If lenDtaqData = 0 and bLiveHldEnabled;
  // timed out with no DTAQ entry — sweep for anything the live
  // notification path can't see (HELD, SAV, etc.)
  For x = 1 to iOutQueues;
    sendAllSplf( outQueueDS(x).outQueue : outQueueDS(x).outQLib );
  EndFor;
  Iter;   // go back to $dataq() rather than falling into the dqtype checks belo
EndIf;
```

This reuses `SendAllSplf()` completely unmodified — no new spool-file-processing code path, so no new correctness risk on top of what IDOCS-147 already validated. The existing `*TERM` / `*SPOOL` handling below is untouched and only runs when `dataq()` actually returned real data.

Why this is safe for the other ~100 engines on the box

- `OQHLDLIVE` defaults to N/blank ⇒ `bLiveHldEnabled` stays `*OFF` ⇒ `@@wait` stays `-1` ⇒ every existing engine's loop behaves byte-for-byte as it does today. The new branch in the receive loop is dead code for them (the `and bLiveHldEnabled` guard).
- `$polling()`-mode engines (`PRDMASTER.DTAQ` blank) are entirely untouched — the whole change lives inside the `Else` (DTAQ-mode) branch.

Duplicate-processing risk — already handled by existing behavior

A periodic re-sweep re-lists *everything* currently in the queue on each timeout, same as the startup sweep does today. This is only safe if a spool file gets removed from the originating queue once processed. It does: `PRR0100`'s existing archive/hold/delete logic (`e_PPARCDEL` — Hold/Archive/Delete, see PRR0100.RPGLE ~2996-3018 region and the IDOCS-127 rewrite above it) already moves, holds elsewhere, or deletes the originating file after `processSplFile()` runs. The periodic sweep relies on exactly the same property the one-shot startup sweep already relies on — not a new assumption.

What was actually implemented (tag 60825, 2026-08-21)

- `PRDOUTQUE.PF` : added `OQHLDLIVE` 1A field, with the `ALTER TABLE` / `LABEL ON COLUMN` template comment mirroring how `PRDTARGET.SPLSTS` was added for IDOCS-147.
- `PRD0055.DSPF` : added a "Process / HLD Live" subfile column (`HLDLIVE` 1A at row 12 col 49) next to the existing "Process at / Start-up" column, plus header text at row 10-11 col 45.
- `PRR0055.SQLRPGLE` : mirrored every place `HldSpl` / `OqHldSpl` is handled — default-if-blank validation, native-I/O save (both the update and write branches), and the embedded-SQL subfile load (both the cursor SELECT and the FETCH).
- `PRR0459.RPGLE` :
  - New global `bLiveHld` (1b0, matches `bProcess`'s declaration width/column exactly — do NOT use a longer name here without re-verifying column alignment; RPG D-spec column alignment is unforgiving of even one extra trailing space).
  - In the outqueue-load loop: sets `bLiveHld = TRUE` if any monitored outqueue for this engine has `o_oqhldlive = 'Y'`.
  - `@@wait` bounded to 30s (only when `bLiveHld`) instead of the original `-1` (indefinite) before the DTAQ receive loop.
  - After each `$dataq()` call in the loop: `if lenDtaqData = 0 and bLiveHld` (a genuine timeout, not a real *SPOOL/*TERM message) → sweep every one of this engine's outqueues via the existing unmodified `sendAllSplf()`, then `iter` to skip the dqtype checks and loop back to waiting.
  - Every new line is tagged `60825` for easy grep/removal if this needs reverting.

Untouched: `$polling()` mode, and every engine that doesn't set `OQHLDLIVE='Y'` on at least one of its outqueues — confirmed by re-reading the diff, `bLiveHld` starts `FALSE` and nothing sets it without that flag.

Deploy steps

1. ✓ Set `PRDOUTQUE.OQHLDLIVE = 'Y'` on the `TONSRENTER` / `TONSRIN` and `TONSRENTER` / `TONSRIN2` rows (done 2026-08-28, via SR Outqueue Maintenance — PRD0055/PRR0055, not raw SQL, so the maintenance program's own save logic was exercised too).
2. ✓ Compile order followed: `PRDOUTQUE.PF` (DDS, `CRTPF` / `CHGPF`) → `PRD0055.DSPF` (`CRTDSPF`) → `PRR0055.SQLRPGLE` → `PRR0459.RPGLE`. `PRR0459` binds `PRR0100` per its own H-spec (`bnddir('PRR0100')`, PRR0459.RPGLE:65) — that binding directory resolved cleanly; no binder issue this time. Compiled into library `IDOCS6` (confirmed to be TONSRENTER's actual runtime library — its active job's library list includes `IDOCS6`, not `PREPROC300`, which turned out to be a stale, unused 2009-era library not in that list at all).

   Two real compile errors surfaced and were fixed along the way (both `RNF7423` "A relational expression or indicator is expected for logical operations" — a bare numeric flag can't be used directly as a boolean in RPG; needed explicit `= TRUE`, matching the existing `bProcess` idiom elsewhere in the same file): line 567 (`If bLiveHld;` → `If bLiveHld = TRUE;`) and line 580 (`and bLiveHld;` → `and bLiveHld = TRUE;`). Also, recompiling `PRDOUTQUE.PF` (a *FILE with real data) required renaming the existing object first — CMS's compile option silently skips a *FILE recompile when the target already exists and holds data. Separately, this same `IDOCS6/PRDOUTQUE` schema change broke an unrelated program, `PRR0010` (an `INSERT INTO PrdOutQue ... SELECT` with an implicit column list whose value count no longer matched the now-5-column table) — fixed by adding `oqHldLive` to its SELECT list.

3. 30-second sweep interval is a starting guess, not confirmed with Orlando — revisit if detection latency matters for the real workload.
4. ✓ Test plan executed 2026-08-28: placed a HLD spool file into TONSRIN2 while the engine was already running and idle (not at startup) — exactly the scenario that failed 3/3 times before this fix. Confirmed picked up and delivered to TONTOIDOCS still `HELD`, without restarting the engine (the IDOCS-147 `PRR0100` fix handles preserving that status). Verified via the delivered spool file's own attributes (Job: `TONSRENTER`, Status: `HELD`, Output queue: `TONTOIDOCS`), not just visual inspection.

Objects changed / compiled for this project

All in library `IDOCS6`.

Object	Type	Attribute
`PRDTARGET`	*FILE	PF-DTA
`PRR0100`	*PGM	RPGLE
`PRDOUTQUE`	*FILE	PF-DTA
`PRD0055`	*FILE	DSPF
`PRR0055`	*PGM	SQLRPGLE
`PRR0459`	*PGM	RPGLE
`PRR0010`	*PGM	SQLRPGLE

`PRDTARGET` and `PRR0100` — Part 1 (Keep Orig Status, steps 1–7). `PRDOUTQUE`, `PRD0055`, `PRR0055`, `PRR0459` — Part 2 (Process HLD Live, step 8). `PRR0010` — unrelated program broken as a side effect of the `PRDOUTQUE` schema change (see Deploy steps above) and fixed alongside it.

IDOCS-147 · Smart Router (IDOCS6) · See `DOCS/IDOCS-147/IDOCS-147.html` in this repository for the technical write-up of both parts of this change.