

Ready to Escalate

Support to engineering plays for technical support teams

Free sample: 3 complete plays. From Draft First. Written by a customer success and support lead.

This sample has three complete plays from the book: the five answers to have before you escalate, how to count impact so priority arguments end, and how to update a customer when there's a confirmed bug and no fix date. Every play in the full book follows the same layout: the move, a decision rule, a worked example, the line not to write, the traps, a check before you send, and an agent prompt.

The examples use a made-up product (Northbeam) and made-up customers. No real customer data or log lines appear anywhere.

The full book also has the running case shown sent badly and then well, the engineer's root cause and the clock change trap, worked judgement calls, a measure sheet for bounce and reopen rates, the escalation packet template with a filled-in example, a five answers checklist, a severity worksheet to agree with engineering, customer messages for every stage, intermittent bug capture instructions, a ping-pong review sheet, a known issues page template, a runbook template and four longer agent prompts.

The five answers

When a bounce is a question support could have answered, the customer waits a day for nothing. Your ping-pong review (Play 22) shows which questions; answer those up front. The five answers are the readiness gate.

SITUATION	Engineering keeps sending tickets back with basic questions, and customers wait a day for each round trip.
TRIGGER	Before every escalation to engineering, no exceptions.

The move

1. Write down the five answers: what the user did, the exact error or wrong result, the impact, whether you can reproduce it, and the environment.
2. Add a sixth line: what you've already ruled out (Play 4).
3. For any answer you don't have, write [MISSING] and get it from the customer or your own testing before escalating.
4. If you truly can't get one, say so and why. A known gap is fine; a hidden one isn't.

Decision rule. Can't answer all five or explain the gap? Not ready; the hold-or-send threshold decides. Security or data issues follow your company's security policy (Play 14).

Example (running case, as of 14 Oct)

Did: Scheduled the "Open orders" export nightly at 02:00 Berlin time.

Error: Export email arrives with "Export failed: ERR_EXPORT_TIMEOUT". No file attached.

Impact: 1 account confirmed (Halvard Supply, acct_4471), 4 failures in 9 nights. Warehouse team starts without the order list.

Repro: Not yet on the live account; manual export of the same report works. Test account repro under way (Play 2).

Environment: Web app, EU region, Business plan, report has 38 columns.

Ruled out: email delivery (fails in our export log, before the mail queue); permissions (same user exports manually).

Don't write

"Customer says exports are broken, please investigate ASAP."

Watch for

- Filling the five with filler ("environment: web") to get past the gate. Specifics or [MISSING].
- Treating the gate as support's homework only. Agree the five with engineering.

If it still stalls: If the customer can't provide an answer, escalate with the gap stated and what you tried.

Agent draft prompt

From this ticket thread: [paste, secrets removed], fill in the five answers (did, error, impact, repro, environment) and a "ruled out" line. Quote errors exactly. Take facts only from the thread; write [MISSING] for anything not there. Then list what to ask the customer, as a numbered list. Don't file anything. Draft only.

Check before you send: Could an engineer start without asking you one question? If not, does the hold-or-send threshold say hold?

Impact counted honestly

When support's impact numbers are inflated, engineering learns to discount them, and the next real fire waits. Counting by ID, keeping confirmed apart from possible, and showing your method builds the trust that gets bugs prioritized.

SITUATION	An argument about whether a bug matters, or a bug with several reports that may or may not be the same.
TRIGGER	Any escalation with more than one report, or any time someone asks "how many?"

The move

1. List confirmed tickets and accounts by ID: same error, same steps, verified by you.
2. List possible matches separately, with why they might be the same.
3. Say how you counted: which search, which date range, which queue.
4. Describe the effect on the customer in one line, and whether a workaround exists and what it costs.
5. Don't round up, don't extrapolate to "probably hundreds".

Decision rule. Only confirmed, ID-linked tickets and accounts go in the count. Everything else goes under "possible" with the reason. If someone asks for a bigger number, give them the method instead.

Example (running case, 15 Oct, after Play 15)

Confirmed: 3 tickets, 2 accounts. #48213 and #48390 (Halvard Supply, acct_4471); #48455 (Brightwater Foods, acct_5120).

Possible: 2 tickets mention export emails "not arriving" with no error quoted: #48107, #48466. Asked both customers for the failure email.

How counted: ticket search "ERR_EXPORT_TIMEOUT" and "export failed", all queues, 1 to 15 Oct.

Effect: nightly order list missing; warehouse starts late. **Workaround:** manual export each morning, a few minutes of someone's time daily (customer's estimate).

Don't write

"This is affecting a lot of customers."

Watch for

- Counting one customer's three tickets as three customers.
- Dropping "possible" matches entirely. They may become confirmed.

If it still stalls: If engineering disputes the count, walk them through the search and the IDs. If they find a mistake, correct it in the ticket openly.

Agent draft prompt

From this list of tickets: [paste IDs, accounts and snippets], sort them into Confirmed (same error text and steps) and Possible (similar, unverified), with the reason for each Possible. Count tickets and distinct accounts separately. Don't count any account twice. Draft only.

Check before you send: Is every number tied to a ticket ID, with "possible" kept apart and the search stated?

Customer updates with no ETA

Without a fix date, the temptation is to go quiet until there's news, or to guess "next release". Scheduled updates that say what's known, what to do meanwhile and when you'll write next keep the customer informed without inventing anything.

SITUATION	The bug is confirmed, there's no fix date, and the customer keeps asking.
TRIGGER	Any confirmed bug on a customer's open ticket. Set the schedule the day it's confirmed.

The move

1. Agree an update rhythm with the customer, based on severity and their needs (for example, twice a week). Put it in your calendar.
2. Each update says: what's known now, what's changed since last time (or "no change"), the workaround, and the next update time.
3. Use plain language about engineering's progress: "reproduced", "cause found", "fix being tested". Only share stages that are true.
4. Never pass on a date engineering hasn't confirmed in writing.

Decision rule. The day a bug is confirmed, give the customer a next update time and keep it, news or not. A date for the fix is only ever engineering's, in writing.

Example (running case, update on Thu 15 Oct)

"Hi Marta, here's today's update on the failing nightly exports. What's new: our engineers have reproduced the failure on their side, which is the step that lets them work on a fix. What hasn't changed: there's no fix date yet, and I won't guess one. Meanwhile, the manual export from the Reports page still works. I'll write again on Monday by 12:00 your time, sooner if anything changes."

Don't write

"It should be fixed in the next release."

Watch for

- "No change" updates that sound like a brush-off. Restate the workaround and the next time.
- Stopping updates after the third one. The calendar reminder is the play.

If it still stalls: If the customer says the updates aren't enough, ask what they need to plan around (a deadline, a meeting) and tell the bug owner about that specific need.

Agent draft prompt

Draft a customer update for a confirmed bug: [summary]. Include what's new since [last update date]: [notes, or "no change"], this workaround: [steps], and the next update time: [time and time zone]. Plain words, no fix dates, no internal names or links. Draft only.

Check before you send: Is there a next update time, and is any fix date engineering's, in writing?

Want the rest?

The full Ready to Escalate playbook has all 24 plays across five chapters, from before you escalate to closing the loop, plus every template and customer message in the back of the book. It's a PDF you can print or keep open beside your tracker. All sales are final, so this sample is here to help you decide.

Questions: hello@draftfirstcs.com

The full playbook has 24 plays

Chapter 1. Before you escalate	1. The five answers; 2. Reproduce from a known state; 3. Quote, don't paraphrase; 4. Rule things out on paper; 5. Hypothesis, labeled
Chapter 2. Investigating hard cases	6. Timeline first; 7. What changed?; 8. Intermittent bug data plan; 9. Narrow the blast radius; 10. Reading logs with an agent
Chapter 3. Writing the escalation	11. Titles engineers can search; 12. Impact counted honestly; 13. Severity without drama; 14. Security and data issues; 15. Duplicate hunting
Chapter 4. While engineering works	16. Customer updates with no ETA; 17. The workaround note; 18. When engineering goes quiet; 19. When the answer is "won't fix"
Chapter 5. Closing the loop	20. Fix shipped, tell everyone; 21. Verify with the customer; 22. The ping-pong review; 23. Known issues page; 24. From ticket to runbook