

Single-Task Agentic AI for Bank Dispute Resolution: A Research and Economic Perspective

Short paper · 25 September 2026

Abstract

A narrow AI agent built for one bank process — card and payment dispute resolution — can deliver most of the value of automation while carrying far less risk than a general-purpose assistant. This paper argues the case from two angles. From research, narrow agents are easier to train, evaluate and govern because the task, its data and its success criteria are well defined. From economics, disputes are high-volume, rules-driven and deadline-bound, so automating the routine majority while routing exceptions to humans lowers cost per case, shortens resolution time and reduces regulatory penalties. We outline how to train such an agent from the bank's historical dispute records and its written methods and procedures, how it runs a case from intake to closure — including asking customers and merchants for clarification and evidence — and where human judgement must stay in the loop.

1. Introduction: the case for narrow agents

Banks do not need an AI that can do everything; they need one that does a specific job reliably, auditably and within the law. Dispute resolution is a strong candidate. A customer reports an unrecognised charge, a duplicate payment, goods not received or a failed ATM withdrawal. The bank must log it, classify it, gather facts, apply network and regulatory rules, decide, communicate and close — often within fixed legal deadlines.

Today much of this is manual. Analysts read free-text complaints, look up transactions across several systems, email customers and merchants, and apply procedure manuals from memory. The work is repetitive, but mistakes are expensive: missed deadlines, wrong outcomes and poor customer experience all carry cost.

A single-task agent targets exactly this gap. It is "sufficient" in a precise sense: capable enough to complete the standard case end to end, and designed to recognise when a case is not standard and hand it to a person.

2. Research perspective: why narrow works better

Narrow agents succeed for reasons that are well understood in machine learning and systems research.

- **Bounded task, measurable success.** Dispute outcomes are labelled in history (upheld, declined, partial refund, chargeback won or lost). This gives a clear target

for training and a clear metric for evaluation, unlike open-ended assistants.

- **Distribution match.** The agent is trained and tested on the same kind of cases it will see in production. Errors are more predictable and drift is easier to detect.
- **Grounding reduces hallucination.** The agent reasons over retrieved facts — the transaction record, the customer's statement, the procedure step — rather than open knowledge. Every decision can cite its evidence.
- **Constrained action space.** It can only call a small set of tools: look up transactions, request documents, issue provisional credit, raise a chargeback, send a letter. A small, permissioned toolset is far easier to secure and test.
- **Calibrated deferral.** Research on selective prediction shows that a model allowed to abstain on low-confidence cases can reach much higher accuracy on the cases it keeps. Human exception handling is the operational form of this idea.
- **Auditability.** A fixed workflow produces a structured case log, which supports model-risk review, regulators and internal audit.

3. Training through data engineering

The agent learns from two sources the bank already owns: its **historical dispute database** (what happened) and its **methods and procedures** (what should happen). Data engineering turns both into usable training and grounding material.

Source	What it provides	Engineering work
Historical dispute cases	Claims, transactions, messages, evidence, decisions, outcomes	Extract, join across systems, de-duplicate, mask personal data, label outcome and reason codes
Analyst notes and correspondence	How staff asked for and weighed evidence	Clean free text, link to case stage, tag "request", "clarification", "decision" steps
Methods and procedures (M&P)	Rules, decision trees, deadlines, letter templates	Convert to structured rules and a retrievable knowledge base with version dates
Card network and regulatory rules	Reason codes, time limits, liability rules	Encode as hard constraints the agent cannot override
Exception and appeal history	Where cases went wrong or were escalated	Use to train the escalation trigger and to build test sets

Three practices matter most:

1. **Clean labels.** Past decisions are not always correct. Reversed or appealed cases should be relabelled, or the agent will learn old mistakes.
2. **Procedure first, history second.** Where history conflicts with current M&P (for example after a policy change), current policy wins. Cases are weighted by recency and tagged with the procedure version in force.
3. **Retrieval plus fine-tuning.** Procedures change often, so they are best retrieved at run time rather than baked into model weights. Fine-tuning or examples from history teach style, classification and when to ask for more evidence.

Before go-live the agent is evaluated on a held-out set of past cases, compared with actual analyst decisions, and run in "shadow mode" — deciding in parallel with humans without acting — until agreement and error rates meet agreed thresholds.

4. End-to-end workflow: intake to closure

The agent owns the whole case, not just one step. Its defining feature is that it can pause, ask for what is missing and resume when it arrives.

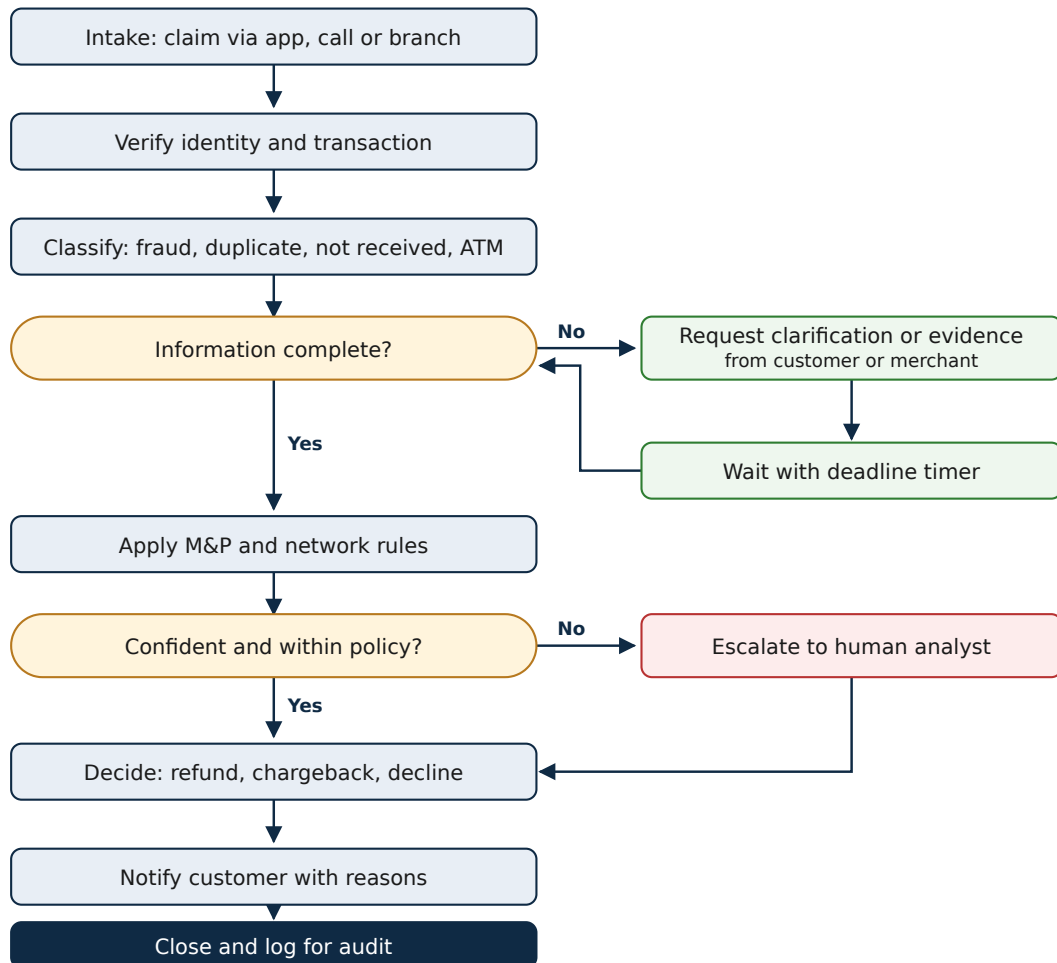


Figure 1. Dispute case flow: the agent loops for missing information and escalates exceptions to a human.

1. **Intake.** Capture the claim from any channel and turn free text into structured fields: amount, date, merchant, reason.
2. **Verification.** Match the claim to the transaction, check the account, and check for duplicate or earlier disputes.
3. **Classification.** Assign a dispute type and network reason code, which sets the rules and deadlines that apply.
4. **Clarification and evidence.** If facts are missing, the agent asks targeted questions — for example, "Did you contact the merchant?" or "Please upload the cancellation email." It can also request a receipt, delivery proof or CCTV log from the merchant or another team. Each request carries a deadline and an automatic reminder.

5. **Provisional action.** Where regulation requires it, the agent issues provisional credit within the legal window while the case continues.
6. **Decision.** It applies procedure to the evidence, drafts a reasoned decision and cites the rule and documents it relied on.
7. **Communication and closure.** It sends the outcome letter, files any chargeback, updates the ledger and closes the case with a full audit trail.

The clarification loop is where most value lies. In manual processes, cases often stall waiting for documents. An agent that asks precisely, tracks every open request and follows up on time removes much of that idle time.

5. Human-in-the-loop for exceptions

The human is not a fallback for failure; they are part of the design. The agent escalates when any of these triggers fires:

- Confidence in the classification or decision is below a set threshold.
- The amount exceeds a value limit set by the bank's risk appetite.
- Signs of first-party fraud, account takeover or money laundering appear.
- The customer is flagged as vulnerable, or complains about the process itself.
- Evidence conflicts, or the case fits no procedure.
- The customer asks to speak to a person.

The analyst receives a prepared case file: a summary, the evidence gathered, the rules checked and the agent's suggested outcome with its reasons. This turns a 30-minute investigation into a short review. Each human decision, especially an override, is fed back as new labelled data, so the escalation rate should fall over time without the agent taking on cases it cannot handle.

A sensible rollout moves in stages: shadow mode, then the agent acting on low-value and simple cases, then wider scope as measured accuracy allows.

6. Economic perspective

The economic case rests on a simple split: automate the routine majority cheaply, and spend expensive human time only where judgement adds value. This is the classic division of labour, with the agent and the analyst each doing what they have a comparative advantage in.

Where the value comes from

- **Lower cost per case.** An agent's marginal cost per dispute is small and falls with volume, while manual cost scales almost linearly with headcount.
- **Faster resolution.** Less idle time waiting for documents means fewer days of provisional credit outstanding and faster recovery through chargebacks.
- **Fewer penalties and losses.** Missed network deadlines forfeit chargeback rights; missed regulatory deadlines bring fines and remediation. Automated timers cut both.

- **Elastic capacity.** Dispute volumes spike after fraud waves, outages or holidays. An agent absorbs peaks without overtime or temporary hiring.
- **Customer retention.** Quick, clear outcomes reduce complaints and churn, which is valuable but harder to measure.
- **Better use of people.** Analysts move from data gathering to complex fraud, vulnerable customers and quality review.

A simple cost model

Let N be annual cases, a the share the agent closes alone, C_h the manual cost per case, C_a the agent's cost per case, C_r the cost of human review of an escalated case, and F the fixed annual cost of building and running the system.

$$\text{Annual saving} = N \cdot [C_h - (C_a + (1 - a) \cdot C_r)] - F$$

Illustrative only (assumed figures, not bank data): with $N = 100,000$, $C_h = \$40$, $C_a = \$3$, $C_r = \$15$, $a = 0.7$ and $F = \$1,000,000$, the new cost per case is about \$7.50 and the net saving is about \$2.25 million a year.

Why not automate 100%? Returns diminish in the tail. The last 10–30% of cases are rare, varied and high-risk, so building automation for them costs more than it saves, and an error there is costly. Keeping humans on exceptions is the economically efficient point, not a compromise. Because the model is narrow, the bank also avoids the much higher cost and risk of building and governing a general-purpose system.

7. Risks, governance and regulation

Risk	Mitigation
Learning biased or wrong past decisions	Relabel appealed cases; test outcomes across customer groups for fairness
Outdated procedures	Retrieve M&P at run time with version control; re-test after each policy change
Hallucinated facts or rules	Require every decision to cite a record or rule; block actions without evidence
Fraudsters gaming the agent	Fraud-signal triggers force escalation; rate limits and anomaly monitoring
Privacy and data leakage	Mask personal data in training; least-privilege tool access; data stays in bank systems
Model drift	Monitor accuracy, escalation rate and overturn rate monthly; retrain on fresh cases

The agent must work within the bank's model-risk framework and applicable consumer-protection rules. In the US, for example, Regulation E and Regulation Z set investigation and response timelines for electronic transfers and card billing errors; in the EU and UK, payment-services rules set refund duties for unauthorised payments. These deadlines should be hard-coded constraints. Customers should be told when AI is involved and always have a route to a human decision.

8. Conclusion

A single-task dispute agent is a practical, lower-risk path to real AI value in banking. It is trained on the bank's own history and grounded in its current procedures. It runs cases from intake to closure, including the clarification and evidence loop where manual processes stall. And it hands genuine exceptions to people with the case already prepared.

Research favours it because narrow tasks are measurable, groundable and auditable. Economics favours it because it captures most of the savings from the routine majority, while leaving the costly, uncertain tail to human judgement. The same pattern — one well-defined process, one agent, humans on exceptions — can then be repeated for other bank operations such as account servicing, KYC refresh or loan document checks.