Technical Case: Argus - Automated Financial Reconciliation & Support Agent for B2C&B2B

Enterprise Multi-Agent Financial Reconciliation System

---

1. Executive Summary

Argus is an intelligent multi-agent system designed to automate financial transaction reconciliation and customer query resolution.

Moving beyond simple chatbots, Argus leverages a stateful, event-driven architecture to process complex financial queries. It cross-references unstructured receipt data (OCR) with structured transaction ledgers (XXX) to deliver accurate, context-aware resolutions for both individual consumers (B2C) and corporate partners (B2B).

---

2. The Business Challenge

The client’s financial support operations faced critical scalability bottlenecks:

- **High Volume & Latency:** Support teams were overwhelmed by routine inquiries about transaction discrepancies ("Why is my transfer pending?"), leading to hours-long response times.
- **Manual Reconciliation:** Agents had to manually match PDF receipts against database records, a process prone to human error.
- **Complex B2B Requirements:** Corporate partners required sophisticated reconciliation for bulk disbursements and invoice verification, which standard support workflows could not handle.
- **Lack of Standardization:** Human responses varied significantly in quality, lacking a consistent audit trail for compliance.

---

3. The Solution: Dual-Pipeline Multi-Agent Architecture

I architected **Argus** as a state-based system using **LangGraph** and **FastAPI**. The core innovation is a **Dual-Pipeline Design** that routes queries based on customer segment complexity.

3.1. LangGraph Orchestration (The "Brain")

The system implements a cyclic state machine with self-correcting capabilities:

- **Resolver Agent:** Performs the initial analysis, matching receipt data to transaction logs.
- **QA Agent (The Guardrail):** A dedicated agent that reviews the Resolver's output. If the resolution is inaccurate or hallucinated, the QA agent rejects it and sends feedback to the Resolver, triggering a retry loop (up to 3 attempts).
- **Drafter Agent:** Once QA passes, this agent formats the technical resolution into a consumer-friendly message (B2C) or a formal corporate report (B2B).

3.2. The Dual-Pipeline Strategy

- **B2C Pipeline (Deterministic):** Optimized for speed and low cost. Uses a linear, highly structured flow to resolve standard transaction queries (e.g., "Where is my money?").
- **B2B Pipeline (Skill-Based):** Optimized for flexibility. Uses a **Semantic Router** to classify complex requests (e.g., "Reconcile this bulk disbursement") and dynamically loads specialized "Skills" (prompts/logic) from a registry.

3.3. LLMOps & Observability

- **Langfuse Integration:** Centralized management of system prompts allows for A/B testing of agent behavior without code deployments. Full tracing provides end-to-end visibility into the multi-agent reasoning chain for compliance audits.
- **Resilience:** Integrated json-repair to automatically fix malformed LLM outputs, ensuring 99.9% system stability even when models behave unpredictably.

---

4. Key Capabilities & Impact

Capability	Business Value
Automated Reconciliation	Resolves **70-85%** of routine transaction inquiries without human intervention.
Self-Correcting QA	Built-in feedback loops reduce hallucinations and ensure high-accuracy responses.
Dynamic Skill Loading	The B2B pipeline can be extended with new reconciliation logic (Skills) via configuration, not code.
Auditability	Complete trace history (Input → Agent Thoughts → QA Validation → Output) for every resolution.

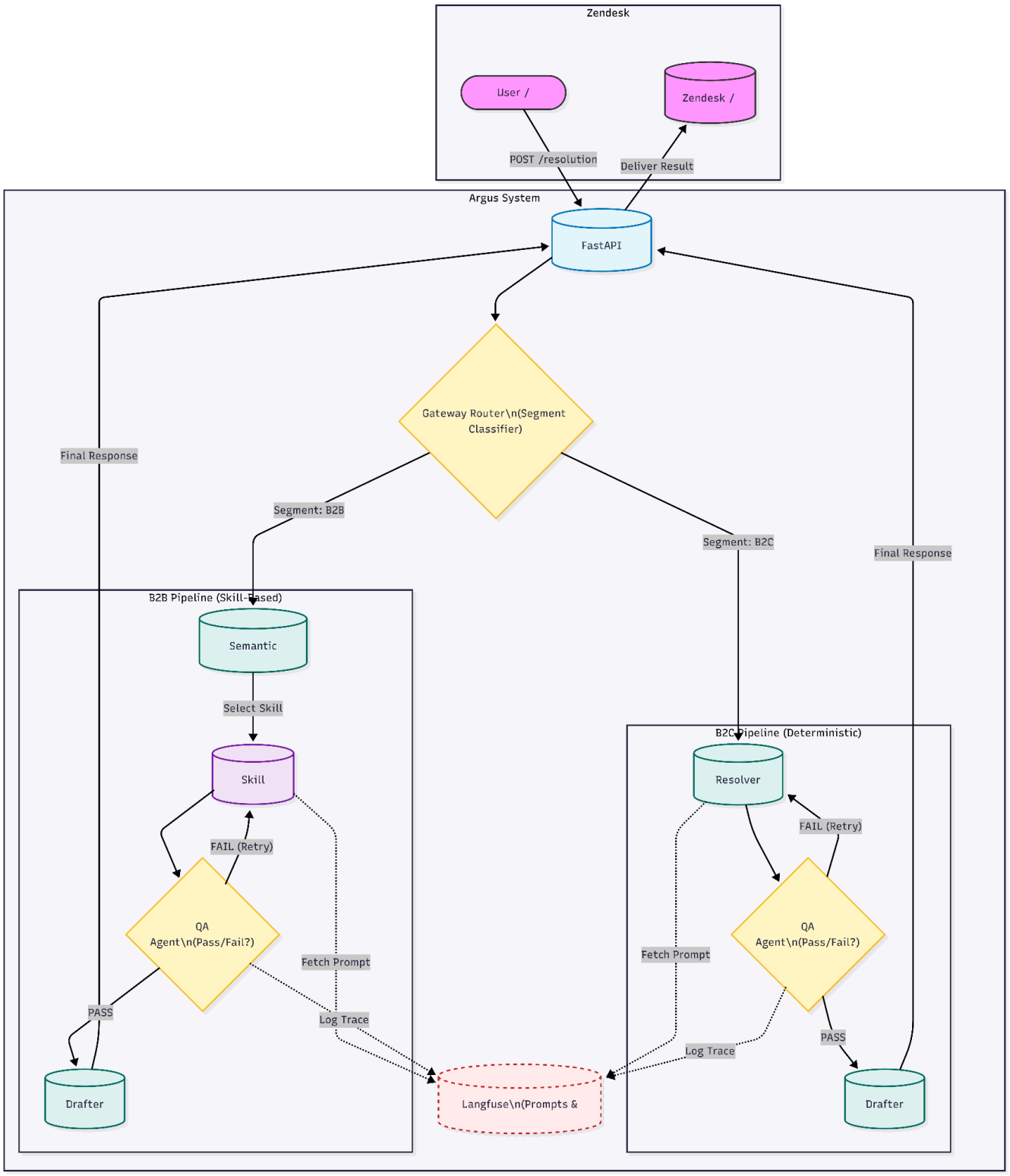
---

5. Technology Stack

- **Orchestration:** LangGraph, LangChain
- **Core Framework:** Python 3.12+, FastAPI, Asyncio
- **AI Models:** Google Gemini (Primary), OpenAI GPT-4, Anthropic Claude
- **Data Validation:** Pydantic V2 (Strict Type Safety)
- **Observability:** Langfuse (Tracing, Prompts, Evals)
- **Utilities:** json-repair, httpx

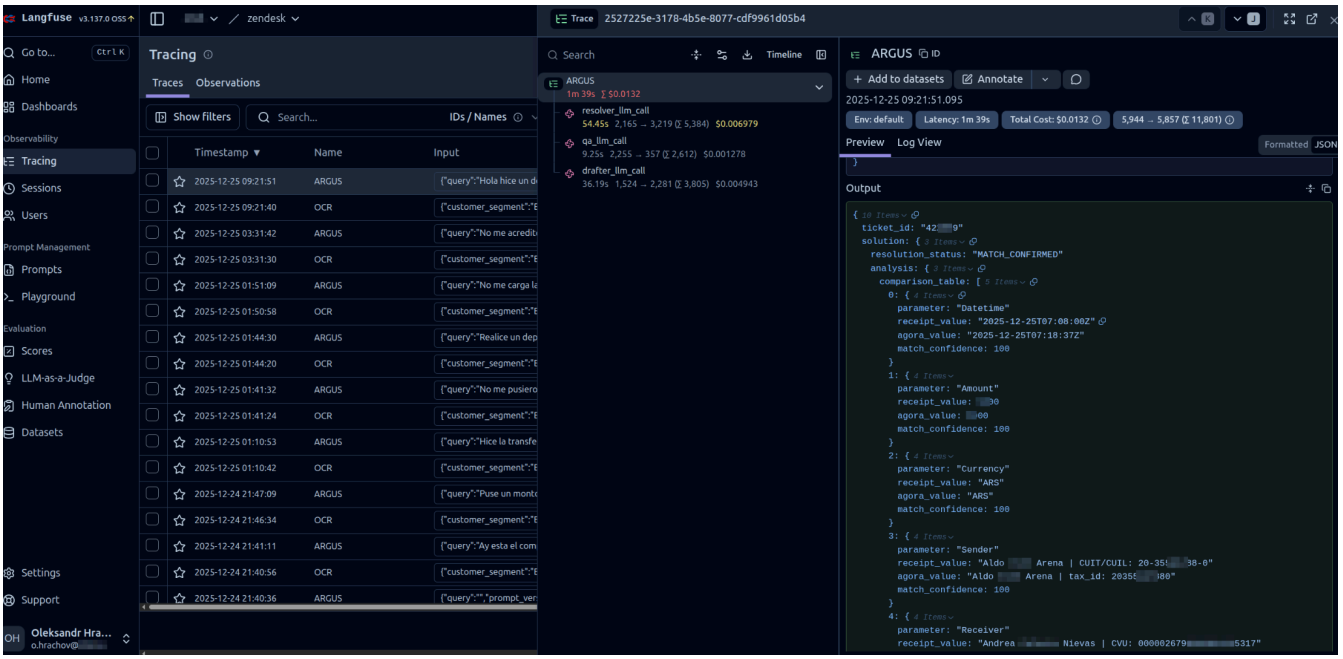
6. System Architecture Flow:

- **Ingestion** (FastAPI): The client (Zendesk) submits a query via a POST endpoint. The API performs strict Pydantic validation on the complex payload (query text, OCR receipt data, and XXX transaction records) and immediately routes the request to the appropriate workflow engine based on the customer segment (B2C vs. B2B).
- **Multi-Agent Orchestration** (LangGraph): A stateful Gateway Router directs the task to one of two dedicated pipelines: a deterministic B2C Graph for standard linear resolution or a B2B Skill Graph for complex dynamic routing. The system orchestrates multiple agents (Resolver, Skill Executor, Drafter) in a cyclic state machine to process the data.
- **LLMOps Integration** (Langfuse):
    - Before Execution: Agents dynamically fetch the latest version of their specific System Prompt (e.g., "skill-b2b-bulk-disbursement") from Langfuse, enabling logic updates without backend deployment.
    - During Execution: The complete multi-agent reasoning chain (Router → Agent Thoughts → LLM Response) is logged asynchronously to Langfuse for full auditability and cost tracking.
- **Validation & Delivery**: Generated resolutions pass through a dedicated QA Agent loop. If the response fails validation, the system triggers an automatic retry. The final output is sanitized via json-repair and stored. The result is then delivered back to Zendesk (via polling or webhook), completing the support automation cycle.


Zendesk
User /
Zendesk /
POST /resolution
Deliver Result
Argus System
FastAPI
Gateway Router\n(Segment Classifier)
Final Response
Segment: B2B
Segment: B2C
Final Response
B2B Pipeline (Skill-Based)
Semantic
Select Skill
Skill
FAIL (Retry)
QA Agent\n(Pass/Fail?)
Fetch Prompt
PASS
Log Trace
Drafter
B2C Pipeline (Deterministic)
Resolver
FAIL (Retry)
QA Agent\n(Pass/Fail?)
Fetch Prompt
Log Trace
PASS
Drafter
Langfuse\n(Prompts &

7. Examples

Langfuse screenshot:



Input:


```
{
   "query": "Hola hice un deposito de XX00 y ya paso bastante y no impacta en mi cuenta",
   "prompt_version": null,
   "receipt_data": {
      "total_pages": 1,
      "pages": [
         {
            "page_number": 1,
            "raw_text": "Comprobante de transferencia\nJueves, 25 de diciembre de 2025 a las 04:08
hs\n\n$ XX00\nMotivo: Varios\n\nDe\nAldoXXXrena\nCUIT/CUIL: 20-3XXX88-0\nMercado Pago\nCVU:
0000003100XXX8269\n\nPara\nAndrea XXX Nievas\nCUIT/CUIL: 27-44XXX36-5\nFIWIND PAY
SA\nCVU: 00000267900XXXX35317\n\nNúmero de operación de Mercado
Pago\n138787XXX79\nCódigo de identificación\n1LMP68NKVYXXXXXR7OEV",
            "language": "es",
            "institution": "Mercado Pago",
            "structured_data": {
               "Header Section": {
                  "Comprobante de transferencia": "",
                  "Fecha": "Jueves, 25 de diciembre de 2025 a las 04:08 hs",
                  "Monto": "$ XX00",
                  "Motivo": "Varios"
               },
               "Sender Section": {
                  "De": "Aldo XXXrena",
                  "CUIT/CUIL": "20-3555XXX-0",
                  "Banco": "Mercado Pago",
```

```
                "CVU": "00000031001XXX8269"
            },
            "Recipient Section": {
                "Para": "Andrea XXXX Nievas",
                "CUIT/CUIL": "27-44XXX36-5",
                "Banco": "FIWIND PAY SA",
                "CVU": "00000267900000XXXX317"
            },
            "Transaction List": [],
            "Footer Section": {
                "Número de operación de Mercado Pago": "13878XXXX79",
                "Código de identificación": "1LMP68NKVYYEXXXXX7OEV"
            }
        },
        "enrichment": {
            "normalized_timestamp": "2025-12-25T04:08:00-03:00",
            "normalized_date": "2025-12-25",
            "normalized_total_amount": XX00,
            "currency_code": "ARS",
            "country_code": "AR",
            "timezone": "America/Argentina/Buenos_Aires",
            "document_type": "receipt",
            "merchant_category_code": "transfers"
        }
    }
  ]
},
"XXXX_data": {
  "provider_transaction_child": {
    "transaction_id": "120XXXX42",
    "amount": XX00,
    "currency": "ARS",
    "timestamp": "2025-12-25T07:18:37",
    "status_code": 10,
    "status_name": "Pending",
    "sender": {
        "sender_name": "Aldo XXX Arena",
        "cvu": null,
        "tax_id": "2035XXXX7880",
        "account_number": null,
        "sender_clabe": "NA",
        "clave_rastreo": null
    },
    "recipient": {
        "recipient_name": "Andrea XXXX Nievas",
        "cvu": "00002679000000XXX317",
        "tax_id": null,
        "account_number": "NA",
        "clabe_number": "NA"
    }
```

```
      }
   },
   "zendesk_metadata": {
      "customer_segment": "B2C",
      "zendesk_ticket_id": 422059,
      "zendesk_ticket_url": "https://XXX.zendesk.com/agent/tickets/42XXX59",
      "transaction_link": "https://XXXX.backXXXb.info/transactions/transactions/1200XXX42/,
      "user_link":
"https://XXXX.backoXXXb.info/transactions/transactions/list?page=1&page_size=50&created_at__gte=
2025-12-25T07:21:51.080Z&created_at__lte=2026-01-01T07:21:51.080Z&external_customer_id=281X
XX0"
   }
}
```


Output:


```
{
   "ticket_id": "42XXX59",
   "solution": {
      "resolution_status": "MATCH_CONFIRMED",
      "analysis": {
         "comparison_table": [
            {
               "parameter": "Datetime",
               "receipt_value": "2025-12-25T07:08:00Z",
               "XXX_value": "2025-12-25T07:18:37Z",
               "match_confidence": 100
            },
            {
               "parameter": "Amount",
               "receipt_value": XX00,
               "XXX_value": XX00,
               "match_confidence": 100
            },
            {
               "parameter": "Currency",
               "receipt_value": "ARS",
               "XXX_value": "ARS",
               "match_confidence": 100
            },
            {
               "parameter": "Sender",
               "receipt_value": "Aldo XXX Arena | CUIT/CUIL: 20-35555XXX-0",
               "XXX_value": "Aldo XXX Arena | tax_id: 20355XXX80",
               "match_confidence": 100
            },
            {
```

```
            "parameter": "Receiver",
            "receipt_value": "Andrea XXX Nievas | CVU: 000002679000000XXX317",
            "XXX_value": "Andrea XXX Nievas | CVU: 000026790000000XXX317",
            "match_confidence": 100
          }
        ],
        "summary_explanation": "All required checks passed with high confidence; receipt matches the XXX transaction.",
        "reasoning": [
          "Step 1: Receipt existence confirmed.",
          "Step 2: Required receipt fields (normalized_total_amount, normalized_timestamp) present.",
          "Step 3: XXX record present for comparison.",
          "P1 Datetime: Receipt UTC 2025-12-25T07:08:00Z vs XXX UTC 2025-12-25T07:18:37Z → Delta ≈ 10 minutes 37 seconds → Score 100.",
          "P2 Amount: Receipt XX00.0 vs XXX XX00.0 → Within tolerance → Score 100.",
          "P3 Currency: Receipt ARS vs XXX ARS → Exact match → Score 100.",
          "P4 Sender: CUIT normalized (receipt 20-355XXX88-0 → 20355XXX880) matches XXX tax_id 20355XXX80 → Unique identifier exact match → Score 100.",
          "P5 Receiver: Recipient name exact match (Andrea XXX Nievas) → Score 100 (CVU shows a formatting/digit-position difference but name match yields full score per rules).",
          "Final: All threshold conditions satisfied → resolution_status set to MATCH_CONFIRMED."
        ]
      },
      "actions": [
        {
          "type": "PUSH_PROVIDER",
          "content": "MATCH_CONFIRMED: Transaction id 12001XXX42 matches provided receipt (operation 1387XXX79). Please proceed with provider-side settlement / release and confirm final posting to the beneficiary account."
        }
      ]
    },
    "qa_status": "PASS",
    "retry_count": 1,
    "feedback": "All checks passed. SimpleResolver output is internally consistent with the protocol.",
    "draft_message": "Hola — gracias por avisarnos. Confirmamos que el depósito de XXX00 ARS que reportaste fue validado y coincide con el registro del proveedor.\n\nQué verificamos (resultados):\n- Monto: XXX00 ARS — coincidencia confirmada.\n- Fecha/hora: las marcas de tiempo coinciden dentro de ~10 minutos — aceptación conforme a las reglas.\n- Moneda: ARS — coincidencia confirmada.\n- Remitente: identificador fiscal verificado y coincide.\n- Beneficiario: nombre coincidente; la CVU mostró una diferencia de formato en la posición de dígitos, pero la regla aplicada considera la correspondencia válida.\n\nSiguientes pasos: Hemos solicitado al proveedor que complete la liquidación y publique el abono en la cuenta beneficiaria. Te avisaremos en cuanto recibamos la confirmación de que los fondos se reflejaron.\n\nSi no ves el abono en tu cuenta en las próximas 24 horas, por favor responde a este mensaje y adjunta la captura del comprobante y la hora aproximada del depósito para que podamos reactivar la gestión.",
    "safety_status": "SAFE",
  }
}
```