Every One Talk deployment, designed the same way — as validated data, not a drawing

A design-and-spec tool for Verizon One Talk call flows: freeform engineer notes parsed into a schema-validated deployment design — rendered as a visual map, a provisioning spec, and a plain-English customer summary.

0
Public intake instruments found — One Talk or any UCaaS platform

10
Routing primitives — the complete One Talk vocabulary

v5
Deployment schema — five iterations to a stable contract

3-tier
Product scope — designed · configured · captured

THE PROBLEM

Solutions engineers designed One Talk call flows in XMind and SimpleMind — drawing tools with no schema and no API. Intake arrived as a form plus freeform shorthand; every engineer documented differently, and the design lived in the picture. A picture can't be validated, diffed, or compiled. The audit made it stranger: no intake instrument exists for One Talk — or practically any UCaaS platform — and Verizon's closest instrument deliberately stores no results. Nobody had built the tool.

WHAT'S BUILT — V1 IN PROGRESS

- A deployment schema as the contract — the complete One Talk routing vocabulary with its constraints and validation rules encoded. A design either validates or it doesn't ship.
- An AI intake parser — freeform engineer shorthand parsed into schema-valid deployment JSON, gated by a golden fixture and a synthetic corpus.
- A visual designer — a React Flow canvas that renders the data; the map, the spec, and the summary all derive from one model.
- Tiered product honesty — One Talk designed, AirDial/fax configured, Contact Center captured for specialist handoff.

> **The safeguard.** The schema is the gate: the parser drafts, the schema decides — fail-closed. What the tool doesn't design, it still captures rather than losing. And the scope ceiling is set by reality: One Talk provisioning is portal-only with no public API, so the tool designs and specs — it never pretends to provision.

THE RESULT

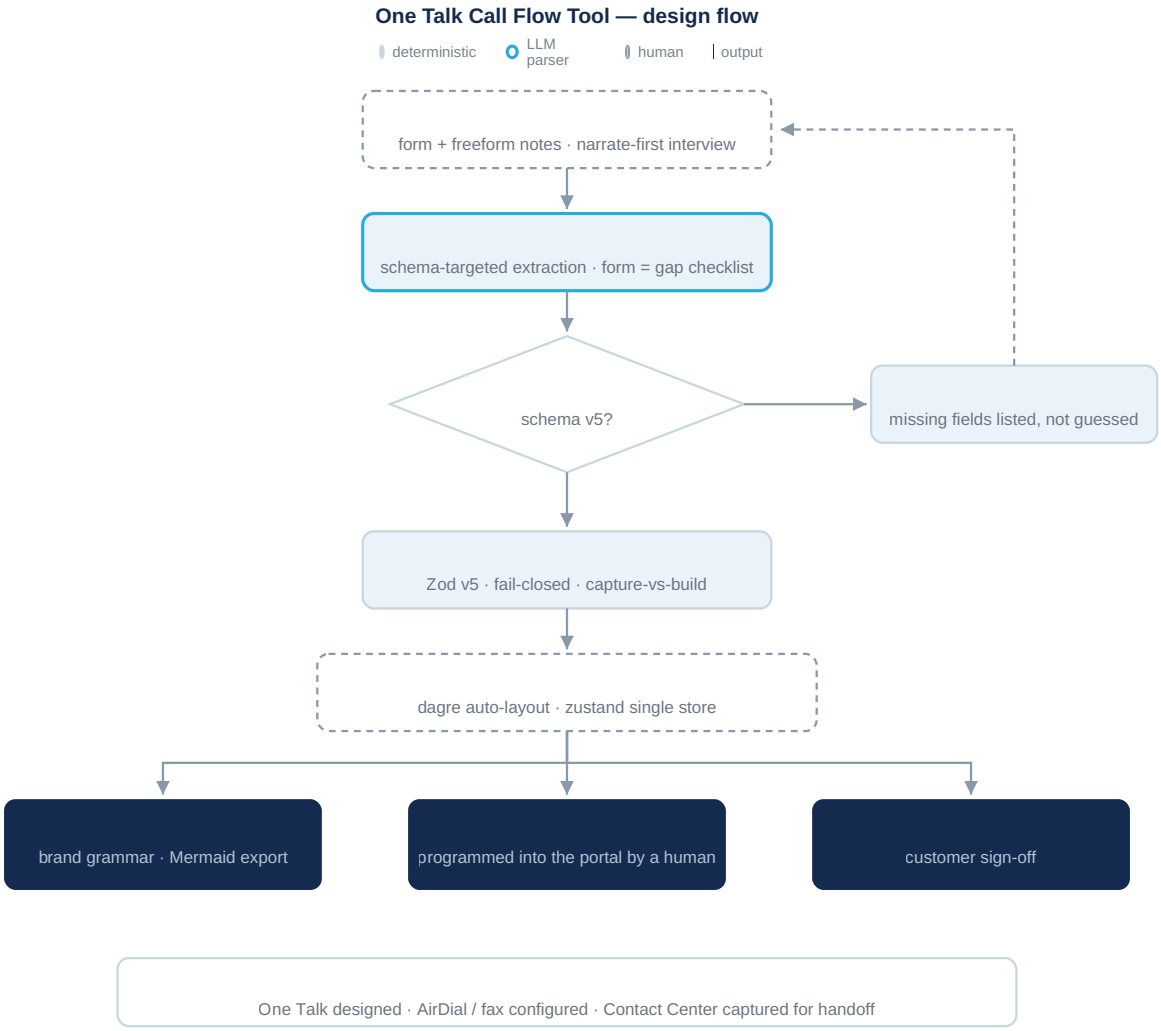
A One Talk deployment now has one canonical representation — validated data — from which the customer map, the programming spec, and the summary all derive. Schema at v5, smoke-tested against a golden fixture and synthetic corpus, briefed to leadership, heading into v1 locked to pure One Talk. SE-pilot metrics land with rollout.

Does your team's most repeated design work live in drawings nobody can validate?

Start with an AI Workflow Teardown →

HOW IT WORKS — DESIGN FLOW

Freeform intake is parsed by an LLM into a deployment model that must validate against the schema before anything renders. The engineer refines the design as data on a visual canvas; the customer-facing map, the provisioning spec, and the plain-English summary are all generated from the same validated model.



The drawing is never the artifact — the map, the spec, and the summary are outputs of one validated deployment model.

Two design choices stand out. **The capture-vs-build doctrine:** requirements outside design scope are captured as data, never silently dropped. **The provisioning boundary:** the platform is portal-only with no public API — the tool outputs a spec for a human to program, and says so.

AI DESIGN — THE SCHEMA IS THE PRODUCT

The defensible idea isn't the canvas — it's the domain model: ten routing primitives (entry numbers, schedules, auto receptionists, hunt groups with all four ring types, simultaneous ring, call queues, lines, voicemail, forwards, announcements) with their real One Talk constraints encoded, hardened across five schema versions of explicit, dated decisions.

Version 4 alone settled: per-user call-forward-no-answer as a first-class field; ring stages that support external cell numbers, with the compile mapping (simultaneous-ring vs. forwarding hops) flagged pending a provisioning answer rather than guessed; main routing split by business hours, after-hours, and holidays; and auto-receptionist no-input handled by convention — the greeting must instruct a key, because the portal offers no timeout route to design around. The schema encodes what One Talk actually does, including its sharp edges — so a design that validates is a design that can be programmed.

RELIABILITY AND GUARDRAILS

MECHANISM	WHAT IT DOES
Fail-closed schema validation (Zod, v5)	Malformed or incomplete designs cannot proceed to outputs
Golden fixture	A known-correct deployment the parser must reproduce — the ground-truth anchor
Synthetic corpus, machine-validated	Three synthetic deployments validated against the schema in a pre-milestone smoke test — passed 2026-07-03
Capture-vs-build doctrine	Requirements outside design scope are captured as data, never silently dropped
Tiered product scope	One Talk designed; AirDial / fax configured; Contact Center captured — the tool declines to design what the business hasn't decided
Provisioning honesty	Portal-only platform, no public API — the tool outputs a spec for a human to program, and says so
Regeneration eval (designed, pending run)	Reconstruct a full dispatch-business archetype from three timestamp narrations alone — an end-to-end test of the parser, not just its parts

INTEGRATION STACK

LAYER	TOOL	ROLE
Frontend	Vite + React + TypeScript	Application shell
Canvas	React Flow + dagre	Interactive call-flow graph with auto-layout
State	zustand	Single source of truth for the deployment model
Validation	Zod (schema v5)	The deployment contract — fail-closed
Intake parsing	LLM parser	Freeform notes → schema-valid deployment JSON
Export	Mermaid	Portable diagram export — a format, not the renderer

WHAT I BUILT — VS. WHAT THE TOOLS DID

- **Market & instrument audit** — verified no public intake instrument exists for One Talk or practically any UCaaS platform; the internal intake form is a strict superset of Verizon's official surface.
- **Domain model + schema** — the ten-primitive One Talk vocabulary with constraints and validation rules, hardened across five versioned, dated decision rounds.
- **Parser + eval design** — the intake-parser prompt, the golden fixture, the synthetic corpus, and the regeneration-eval design.
- **Whiteboarding methodology** — narrate-first interview doctrine, a three-fork topology model (entry, time sensitivity, terminal disposition), and brand visual grammar.
- **Reference archetypes** — single-site; dispatch / service with multi-stage hunts and external cell members; multi-site with central toll-free and per-site E911.
- **Scope governance** — v1 locked to pure One Talk; tiered scope for adjacent products; a ten-page executive brief to leadership.

BUILD-STATE EVIDENCE

EVIDENCE	DETAIL
Market audit	0 public intake instruments found; the internal form is a strict superset of Verizon's official intake surface
Schema maturity	v3 (capture-vs-build doctrine) → v4 (four dated routing decisions) → v5 (tiered scope + per-site metadata: E911, install mode, site-qualification readings)
Pre-milestone smoke test	Golden fixture + three synthetic deployments machine-validated against the schema — passed 2026-07-03
Governance	Ten-page executive brief delivered to leadership, 2026-07-03
Pending	SE-pilot metrics (with v1 rollout); multi-engineer corpus test; regeneration-eval run

TRADEOFFS AND LIMITS

Design-and-spec only — no provisioning. One Talk routing is portal-only with no public API; the ceiling is set by the platform, not the tool. Revisit if Verizon ships a provisioning API.

Staged escalation unresolved. Simultaneous-ring vs. forwarding-hop semantics are flagged in-schema pending a provisioning answer — not guessed.

Contact Center is captured, not designed. The business hasn't selected a platform; a design tool that guesses at unmade business decisions is a liability with good UX.

Parser validated on a fixture + synthetic corpus, not yet a multi-engineer corpus. Real call-note documents from other SEs aren't collected yet; the golden fixture anchors correctness in the meantime.

CONSULTING TAKEAWAY

The real competitor wasn't XMind — it was the idea that the drawing is the artifact. A drawing can't be validated, diffed, or compiled; it can only be looked at, and every engineer looks differently. Model the call flow as data with the platform's real constraints encoded, and the picture becomes the cheapest thing the system produces.

Five schema versions later, the lesson: the schema was never really a technical document. It was five rounds of settling arguments about what a One Talk deployment *is* — arguments that used to be re-fought, silently and differently, inside every engineer's drawing tool.