CASE STUDY · DEEP DIVE EXHIBIT · ~6 MIN

TM Forum Open API Intelligence Suite & Spectra AI Copilot

“When enterprise JSON payloads become unreadable, transform data contracts into observable, interactive graphs with client-side AI.”

ROLE
Creator & Principal Systems Architect

TIMELINE
Architecture & Open Systems Initiative · 2026

DOMAIN
Telco Systems & Applied AI Architecture

KULDEEP SINGH · DIGITAL ARCHITECTURE

PART I · THE CASE

What was built, and why

Architected a browser-local enterprise suite for 23+ TM Forum Open APIs. Combines React Flow DAG visualization, an interactive laser presentation surface, AJV schema compliance, and Spectra AI - a hybrid AI orchestrator pairing in-browser WebGPU local inference (Qwen 3.5) with direct-CORS cloud models.

Modern telecommunications architectures undergoing TM Forum Open Digital Architecture (ODA) transformations rely on over 23 Open APIs (TMF620 Product Catalog, TMF622 Product Ordering, TMF637 Product Inventory, TMF641 Service Ordering, TMF679 Offering Qualification, and related domains). Enterprise payloads in these systems frequently span multi-megabyte, deeply nested JSON documents featuring multi-level product hierarchies, dynamic characteristics, place relationships, and complex pricing elements. Standard text editors and tree viewers become virtually unusable, leading to severe communication friction during architectural reviews and client discovery workshops.

Telco architects, BSS engineers, and solution teams lacked a dedicated visual and analytical medium to inspect, present, and validate complex TMF payloads. Existing generic JSON tools offer no semantic understanding of TM Forum data models, collapse under circular references, offer no schema compliance diagnostics, and lack real-time presentation capabilities. Furthermore, strict enterprise data privacy rules prevent teams from sending sensitive client telecom contracts and quote payloads to third-party cloud AI services for analysis.

CONSTRAINTS

· Must run 100% client-side in the browser with zero mandatory backend server dependencies to protect proprietary enterprise telco data

· Sub-second DAG graph rendering and layout for massive hierarchical order and inventory payloads (up to 10MB JSON)

· Spectra AI must support local in-browser WebGPU inference (zero token cost, zero data leakage) alongside native direct CORS cloud models

· Strict schema validation supporting AJV Draft-07 and 2020-12 across 23+ TM Forum Open API specifications with path-level error diagnostics

· Seamless real-time presentation mode with laser pointer drawing trails to facilitate live architectural brain-storming and client discovery

PART I · ARCHITECTURE

The system as designed

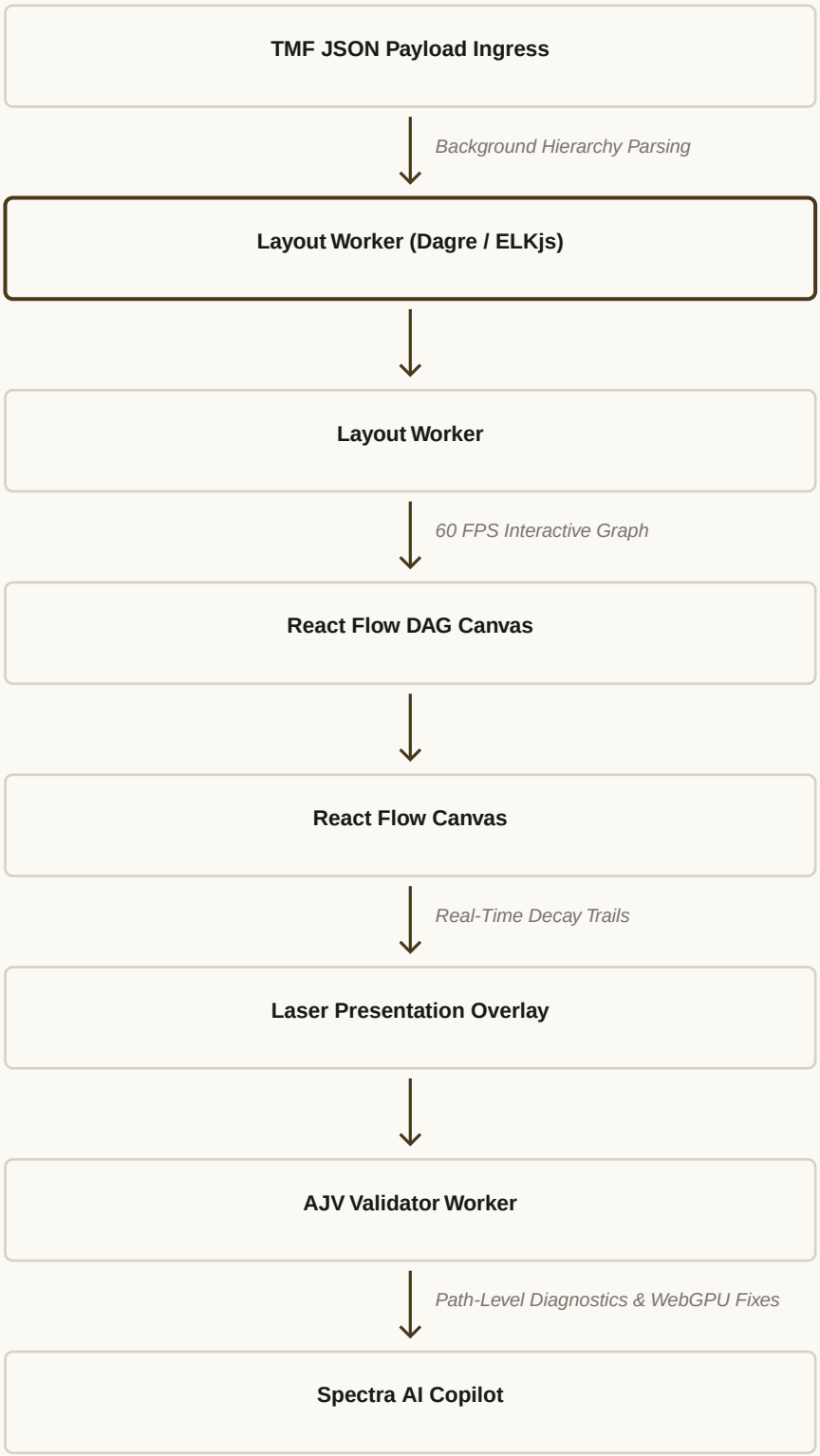
Architected as an ultra-fast, client-side Single Page Application (SPA) pairing React Flow (@xyflow/react) with asynchronous graph layout worker threads (Dagre and ELKjs). Built an interactive presentation overlay with mathematical laser trail decay, a real-time schema diffing engine, and an AJV-powered validator. The intelligence layer, Spectra AI Copilot, is engineered as a hybrid AI orchestrator: users can sign in with Google to execute

KULDEEP SINGH

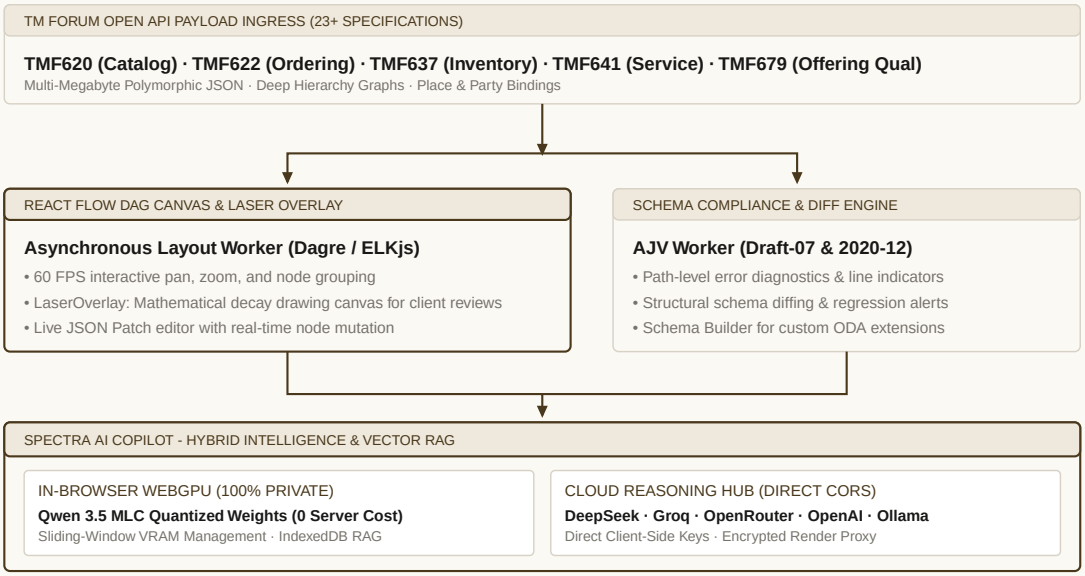
local quantized LLMs (Qwen 3.5 builds) directly on their client GPU via browser WebGPU, or connect directly to cloud reasoning models (DeepSeek, Groq, OpenRouter, OpenAI, and Ollama) with zero backend retention. An IndexedDB vector store enables local RAG over custom user guides with exact specification citations.

KULDEEP SINGH

TM FORUM API VISUALIZER & SPECTRA AI COPILOT ARCHITECTURE



Client-side single-page architecture orchestrating React Flow DAG layout workers, SVG laser drawing canvas, AJV schema compliance, and hybrid WebGPU/Cloud Spectra AI inference.



Full system schematic - TM Forum API Visualizer & Spectra AI Copilot Architecture.

PART I · DECISIONS

Choices, engineering, tradeoffs

React Flow & Directed Acyclic Graph (DAG) Layouts vs. Static Collapsible JSON Trees

Nested JSON trees obscure topological relationships between parent/child order items, billing accounts, and qualifying characteristics.

CHOSEN

React Flow graph canvas orchestrated with Dagre and ELKjs layout engines inside dedicated Web Workers.

WHY

Transforms abstract data contracts into an intuitive visual graph where architects can navigate dependencies, inspect polymorphic attributes, and understand fulfillment topology in real time.

In-Browser WebGPU Local AI + Native CORS Cloud Hub vs. Centralized Heavy AI Server

Enterprise telco payloads contain commercially sensitive pricing, network topologies, and customer identities that cannot touch third-party servers.

CHOSEN

Hybrid AI architecture featuring in-browser WebGPU inference via @mlc-ai/web-llm (Qwen 3.5) paired with native browser CORS for cloud models.

WHY

Provides 100% data privacy with zero server compute costs. Prompts and weights run purely in local browser memory and WebGPU vRAM.

Real-Time Laser Pointer Presentation Canvas vs. Screen Annotation Plugins

Architects frequently conduct live virtual sessions to explain complex payload dependencies to stakeholders without screen clutter.

CHOSEN

Custom SVG pointer drawing surface (LaserOverlay) with fading quadratic bezier trails and 1500ms glow decay.

WHY

Allows presenters to naturally guide client attention directly on active nodes without leaving the browser or installing extra software.

AJV Schema Compliance & Live JSON Patching vs. Read-Only Static Validators

Architects not only need to find errors in payloads; they need to simulate fixes and author schema extensions on the fly.

CHOSEN

Integrated Schema Hub and Schema Builder with path-level JSON patch generation and instantaneous visual feedback.

WHY

Accelerates new TMF API development cycles by allowing engineers to modify payloads and verify compliance live.

ENGINEERING NOTES

· Decoupled layout computation into a dedicated Web Worker (layout.worker.js) utilizing Dagre and ELKjs, keeping the React UI thread at a smooth 60 FPS even when rendering hundreds of node hierarchies.

· Implemented a dynamic local sliding window memory algorithm for WebGPU inference that automatically scales context to protect browser GPU memory and prevent vRAM exhaustion.

· Engineered an SVG-based laser trail renderer with decay timers and pointer capture (laserTrail.js), enabling smooth multi-stroke drawing across zoomable canvas surfaces.

· Constructed a browser-local vector indexing pipeline (docChunker.js, pdfTextExtractor.js, vectorStore.js) using IndexedDB (spectra_chats_db) for local document RAG with API scope tagging.

· Built dual deployment pipelines: static production on Hostinger CDN (tmf.victorkuldeep.com) and Cloudflare Workers (tmf-visualizer.victorkuldeep.workers.dev) with custom worker routing.

PART I · OUTCOMES

Tradeoffs, results, lessons

In-Browser WebGPU Compute vs Cloud API Model Capabilities

Gain: Zero server hosting cost, zero token billing, and total compliance with strict enterprise data residency rules.

Cost: Requires client device with WebGPU support (Chrome/Edge/Brave) and initial download of quantized model weights (~500MB - 1.5GB).

Mitigation: Engineered a dual-engine Model Hub: users without WebGPU can connect lightweight cloud keys (DeepSeek/Groq) with native CORS or run local Ollama/LM Studio.

Full Client-Side Schema Validation vs Server-Side Verification

Gain: Instant sub-millisecond feedback on every keystroke without network latency.

Cost: Requires packaging schemas for 23 TM Forum APIs into client bundle chunks.

Mitigation: Employed dynamic schema registry caching and lazy-loading for API definitions (TMF620-TMF764).

OUTCOMES

· Deployed live to production at tmf.victorkuldeep.com and Cloudflare Workers UAT mirror with global sub-second availability

· Comprehensive visual, structural, and compliance support for 23 TM Forum Open APIs spanning Product, Service, Resource, Customer, and Billing

· Zero server token costs: 100% private in-browser WebGPU AI inference with Qwen 3.5 local models

· Adopted by architects and BSS engineers for interactive client design sessions using the built-in laser pointer presentation tool

· Accelerated schema compliance and payload debugging from hours of manual JSON parsing to instant visual diagnosis

LESSONS

· Data contracts should not be locked in raw text; visual topology reveals structural defects that syntax highlighting cannot.

· Client-side WebGPU and browser-local RAG represent the future of enterprise AI for confidential customer payloads.

· Building specialized, domain-tailored visual tools elevates architectural influence far beyond standard slide decks and static documentation.

01 · ORIENTATION

An exhibit, not an article

Seven layers, each with something to touch. Start anywhere - the path remembers where you are.

You are about to read a telecom payload the way architects wish they could - as a living diagram.

KULDEEP SINGH

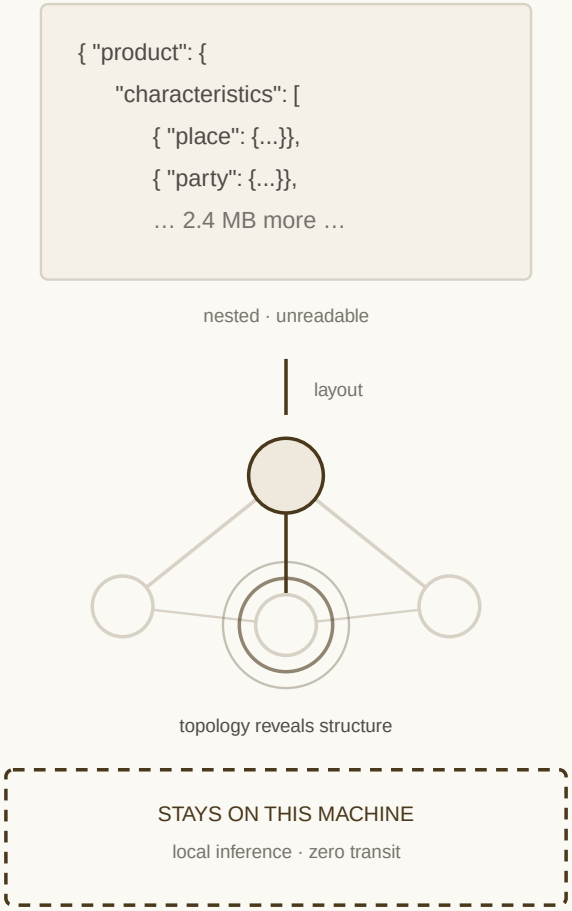


Exhibit plate - the system at a glance.

This exhibit walks through a browser-local suite for 23+ TM Forum Open APIs: multi-megabyte JSON payloads become navigable graphs, a laser pointer draws over live topology, schemas validate path by path, and a hybrid AI copilot explains what you are looking at.

Everything runs client-side. That constraint is the whole design: telco payloads carry commercial sensitivity, so nothing may require a server to function.

How to use this exhibit

- Inspect each layer of the visualization pipeline
- Step one payload from raw JSON to presented insight
- Decide where AI inference should physically run
- Break the payload, then watch validation and layout respond

02 · THE PROBLEM

The unreadable contract

Megabyte payloads, no semantics, no presentation surface, and data that cannot travel.

Start from the unreadable payload, not the tooling.

TM Forum payloads span megabytes of nested product hierarchies, dynamic characteristics, and pricing elements. Generic JSON viewers collapse under circular references, say nothing about TM Forum semantics, and offer no way to present findings to a room.

Worse, the obvious AI answer - paste the payload into a cloud model - violates the data rules of every serious engagement. Sensitive contracts cannot leave the building to be understood.

THE CONSTRAINTS THIS FAILURE IMPOSES

- Must run 100% client-side in the browser with zero mandatory backend server dependencies to protect proprietary enterprise telco data
- Sub-second DAG graph rendering and layout for massive hierarchical order and inventory payloads (up to 10MB JSON)
- Spectra AI must support local in-browser WebGPU inference (zero token cost, zero data leakage) alongside native direct CORS cloud models
- Strict schema validation supporting AJV Draft-07 and 2020-12 across 23+ TM Forum Open API specifications with path-level error diagnostics
- Seamless real-time presentation mode with laser pointer drawing trails to facilitate live architectural brainstorming and client discovery

03 · THE ARCHITECTURE

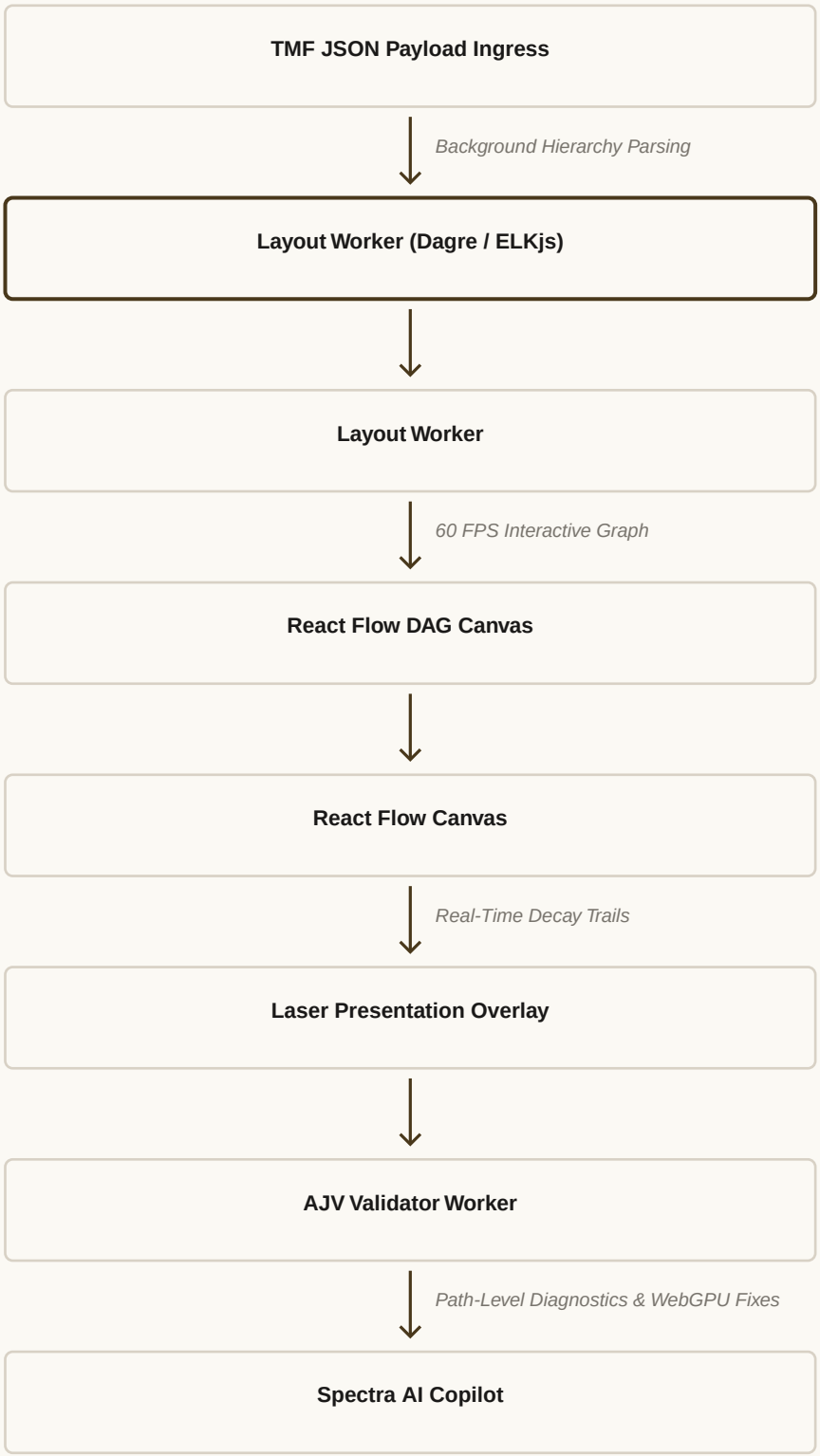
Inspect the system

From pasted JSON to presented, validated, AI-explained topology.

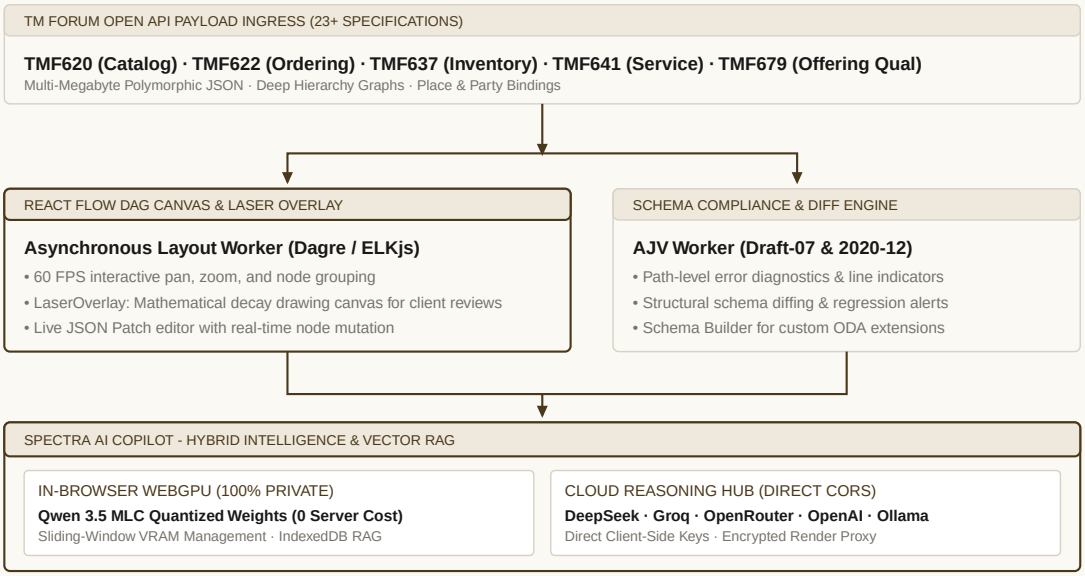
Five layers, all in the browser tab. No server required for any of them.

KULDEEP SINGH

TM FORUM API VISUALIZER & SPECTRA AI COPILOT ARCHITECTURE



Client-side single-page architecture orchestrating React Flow DAG layout workers, SVG laser drawing canvas, AJV schema compliance, and hybrid WebGPU/Cloud Spectra AI inference.



Full system schematic - TM Forum API Visualizer & Spectra AI Copilot Architecture.

Boundary 01 / 06

Payload Ingress

Parse megabytes without fear.

PURPOSE

Accept hostile real-world payloads - huge, nested, circular - and normalize them for layout.

BOUNDARY

It structures data. It renders nothing and judges nothing about correctness.

FAILURE PREVENTED

Tab crashes and frozen renderers on enterprise-scale payloads.

Boundary 02 / 06

Layout Worker

Graphs computed off the UI thread.

PURPOSE

Compute DAG layouts for hundreds of nodes without dropping animation frames.

BOUNDARY

Geometry only. It never touches the DOM or the render loop.

FAILURE PREVENTED

Jank and input lag while large topologies arrange themselves.

Boundary 03 / 06

DAG Canvas

Topology you can navigate.

PURPOSE

Turn abstract contracts into a graph architects can read at a glance.

BOUNDARY

Presentation and navigation. Validation and intelligence live downstream.

FAILURE PREVENTED

Hours lost tracing parent-child fulfillment chains through raw JSON.

Boundary 04 / 06

Laser Presentation

Draw attention, leave no mark.

PURPOSE

Let presenters guide a room's eyes across live topology without leaving the browser.

BOUNDARY

Ephemeral ink only. It annotates nothing permanently and alters no data.

FAILURE PREVENTED

Lost audiences and the context-switching of third-party annotation tools.

Boundary 05 / 06

Schema Compliance

Path-level diagnostics, live.

PURPOSE

Prove payloads against 23+ TM Forum schemas with errors you can act on.

BOUNDARY

It diagnoses and suggests patches. It never silently rewrites payloads.

FAILURE PREVENTED

Integration failures discovered in downstream systems instead of at authoring time.

Boundary 06 / 06

Spectra AI Copilot

Private inference, local or cloud.

PURPOSE

Explain payloads and suggest fixes without the data ever being obligated to leave the machine.

BOUNDARY

It advises on what you show it. Local mode guarantees nothing crosses the network at all.

FAILURE PREVENTED

Sensitive contract data transiting third-party AI services for basic comprehension.

04 · FOLLOW THE FLOW

One payload, five hops

Watch raw text become a graph you can present and question.

Now let's follow one payload from pasted text to explained insight.

01 **Payload Ingress**
A multi-megabyte TM Forum payload is pasted in. Circular references are detected, hierarchy is modeled, the tab survives.

02 **Layout Worker**
The hierarchy ships to a worker thread where Dagre or ELKjs computes positions. The UI thread stays at full frame rate throughout.

03 **DAG Canvas**
The positioned graph renders - navigable, zoomable, inspectable. A structural defect that hid in text is now visibly wrong.

04 **Laser Presentation**
In a live review, the presenter draws fading trails directly over the topology to walk stakeholders through the defect. No plugins, no screenshots.

05 **Schema Compliance**
Validation runs against the matching TM Forum schema and reports the exact paths that violate it, with patch suggestions.

06 **Spectra AI Copilot**
The architect asks what a construct means. A local WebGPU model answers from the payload and cited specs - nothing leaves the machine.

05 · ARCHITECTURE DECISIONS

The choices that hold it up

The recorded decisions, each with the alternative it beat - then your turn.

KULDEEP SINGH

You have seen the pipeline. Here are the recorded decisions behind it.

React Flow & Directed Acyclic Graph (DAG) Layouts vs. Static Collapsible JSON Trees

Nested JSON trees obscure topological relationships between parent/child order items, billing accounts, and qualifying characteristics.

CHOSEN

React Flow graph canvas orchestrated with Dagre and ELKjs layout engines inside dedicated Web Workers.

WHY

Transforms abstract data contracts into an intuitive visual graph where architects can navigate dependencies, inspect polymorphic attributes, and understand fulfillment topology in real time.

ALTERNATIVES CONSIDERED

- Collapsible JSON tree renderers
- Raw code editors with syntax highlighting
- Custom SVG canvas from scratch

In-Browser WebGPU Local AI + Native CORS Cloud Hub vs. Centralized Heavy AI Server

Enterprise telco payloads contain commercially sensitive pricing, network topologies, and customer identities that cannot touch third-party servers.

CHOSEN

Hybrid AI architecture featuring in-browser WebGPU inference via @mlc-ai/web-llm (Qwen 3.5) paired with native browser CORS for cloud models.

WHY

Provides 100% data privacy with zero server compute costs. Prompts and weights run purely in local browser memory and WebGPU vRAM.

ALTERNATIVES CONSIDERED

- Centralized Node.js backend streaming all prompts to OpenAI
- Requiring every user to install local Python environments
- Static rule-based validation scripts only

Real-Time Laser Pointer Presentation Canvas vs. Screen Annotation Plugins

Architects frequently conduct live virtual sessions to explain complex payload dependencies to stakeholders without screen clutter.

CHOSEN

Custom SVG pointer drawing surface (LaserOverlay) with fading quadratic bezier trails and 1500ms glow decay.

WHY

Allows presenters to naturally guide client attention directly on active nodes without leaving the browser or installing extra software.

ALTERNATIVES CONSIDERED

- Third-party zoom annotation tools
- Static screenshot markup tools
- Mouse cursor highlighting

AJV Schema Compliance & Live JSON Patching vs. Read-Only Static Validators

Architects not only need to find errors in payloads; they need to simulate fixes and author schema extensions on the fly.

CHOSEN

Integrated Schema Hub and Schema Builder with path-level JSON patch generation and instantaneous visual feedback.

WHY

Accelerates new TMF API development cycles by allowing engineers to modify payloads and verify compliance live.

ALTERNATIVES CONSIDERED

- Standard post-call JSON linter
- Online schema validator sites
- CLI-only validation scripts

CHALLENGE THE DESIGN

The payload needs AI explanation. Where does inference physically run?

A CENTRAL BACKEND STREAMING TO A CLOUD MODEL

Simplest to build - and disqualified on arrival. Sensitive telco payloads transiting your servers to a third party fails every enterprise review that matters.

NO AI - STATIC RULES AND DOCUMENTATION

Zero privacy risk and zero comprehension help. Payloads stay as unreadable as they started, which was the original complaint.

HYBRID: LOCAL WEBGPU FIRST, DIRECT CLOUD AS FALLBACK

This is the architecture's answer. Data never has to leave the machine, capable hardware gets free private inference, and the cloud path covers the rest without your servers in the middle.

HOW THE ARCHITECT REASONED

Privacy was a placement constraint, not a feature request. Once inference had to live with the data, the hybrid design was nearly forced - and better for it.

Graph layout for hundreds of nodes blocks the UI thread for seconds. What moves?

CAP THE GRAPH SIZE

Protects the frame rate by refusing the actual workload. Enterprise payloads do not shrink because the tool prefers it.

USE A CHEAPER LAYOUT ALGORITHM

Faster and uglier - unreadable topology at full speed is still unreadable. Quality of layout is the product here.

MOVE LAYOUT TO A WEB WORKER

This is the architecture's answer. Full-quality Dagre and ELKjs layouts compute off-thread while the canvas stays interactive at sixty frames.

HOW THE ARCHITECT REASONED

The UI thread belongs to fingers. Anything measured in hundreds of milliseconds belongs in a worker - layout doubly so.

06 · FAILURE MODES

Let's try to break it

Circular payloads, schema drift, and GPU-less machines - each with a landing place.

Now let's try to break it. Each failure below lands somewhere specific - that is the design working.

The circular payload

WHAT GOES WRONG

A payload references itself - the shape that kills naive tree renderers.

WHERE IT LANDS

Ingress detects cycles during modeling and represents them explicitly. The graph renders references as links, not infinite recursion.

The machine without WebGPU

WHAT GOES WRONG

A stakeholder's laptop cannot run local models - no GPU, no weights, no patience.

WHERE IT LANDS

The copilot degrades to direct-CORS cloud models or local Ollama with the same interface. Privacy posture changes explicitly, never silently.

The payload from a newer spec

WHAT GOES WRONG

A partner sends TMF data against a schema revision the validator has never seen.

WHERE IT LANDS

Version-aware AJV packs report precisely what is unknown versus what is wrong, and the schema builder covers custom ODA extensions.

The thousand-node order

WHAT GOES WRONG

An order payload expands into a node count that would freeze main-thread layout.

WHERE IT LANDS

Layout never runs on the UI thread, so the worst case is a progress state on the canvas - never a dead tab.

The payload that must not travel

WHAT GOES WRONG

A contract payload arrives under strict instructions: analyze, but nothing leaves this machine.

WHERE IT LANDS

Local inference plus IndexedDB RAG complete the whole loop in-browser. The architecture treats this as the normal case, not a special mode.

STANDING COMPROMISES

In-Browser WebGPU Compute vs Cloud API Model Capabilities

Gain: Zero server hosting cost, zero token billing, and total compliance with strict enterprise data residency rules.

Cost: Requires client device with WebGPU support (Chrome/Edge/Brave) and initial download of quantized model weights (~500MB - 1.5GB).

Mitigation: Engineered a dual-engine Model Hub: users without WebGPU can connect lightweight cloud keys (DeepSeek/Groq) with native CORS or run local Ollama/LM Studio.

Full Client-Side Schema Validation vs Server-Side Verification

Gain: Instant sub-millisecond feedback on every keystroke without network latency.

Cost: Requires packaging schemas for 23 TM Forum APIs into client bundle chunks.

Mitigation: Employed dynamic schema registry caching and lazy-loading for API definitions (TMF620-TMF764).

07 · REFLECTION

What remains true

Limits, stated plainly - and what changes next.

You have seen the pipeline, the decisions, and the failure boundaries. Two honest notes before you leave.

This suite is strongest where payloads are large, structured, and sensitive: exactly the workloads generic tools abandon. Visualization plus validation plus private explanation compound - each layer makes the others more valuable.

Its limits are stated, not hidden. Client hardware bounds the local AI story, schema packs must be maintained as specs evolve, and a browser tab is still a browser tab for the truly enormous.

WHAT I WOULD CHANGE TODAY

· Stream layout progressively so thousand-node graphs become navigable before finishing

· Version schema packs as data updates independent of application deploys

· Benchmark local model quality per TM Forum domain and publish the gaps

VERIFIED OUTCOMES

· Deployed live to production at tmf.victorkuldeep.com and Cloudflare Workers UAT mirror with global sub-second availability

· Comprehensive visual, structural, and compliance support for 23 TM Forum Open APIs spanning Product, Service, Resource, Customer, and Billing

· Zero server token costs: 100% private in-browser WebGPU AI inference with Qwen 3.5 local models

· Adopted by architects and BSS engineers for interactive client design sessions using the built-in laser pointer presentation tool

· Accelerated schema compliance and payload debugging from hours of manual JSON parsing to instant visual diagnosis

KULDEEP SINGH

More in the Deep Dive Series

01 · Generic Entity Scoring Engine, Native to the CRM

02 · BrewCMS: Open-Source Content Control Plane & Agent Operating System

03 · Local Evaluation of Salesforce RCA - CML

04 · TM Forum Open API Intelligence Suite & Spectra AI Copilot

05 · WebAssembly in Salesforce LWC: 500K Record Engine & Net-Zero DB

06 · Multi-Site Quoting Without Leaving the CRM

Continue the exploration

Interactive exhibit: victorkuldeep.com/case-studies/tmf-api-visualizer-spectra-ai/deep-dive

Long-form essays: victorkuldeep.com/thinking

Ask AXIOM: victorkuldeep.com/ask

Connect: victorkuldeep.com/connect

Kuldeep Singh

Principal Technical Architect · Salesforce Industries · Revenue Cloud (RCA) · Agentic AI