CONF42 CLOUD NATIVE 2026

Design Systems for Scalable Cloud-Native Analytics Platforms

A seven-layer architectural framework for enterprise analytics design systems

Sonali Priya

Creative Design Director | LB Networks

hello@sonalipriya.com

THE CHALLENGE

Why Analytics Platforms Break at Scale

Enterprise analytics serves diverse users, heterogeneous data, and strict governance - consumer design systems were not built for this.

Fragmentation

Teams duplicate charts and dashboards.
No shared components, no reuse across products.

Visual Drift

Every team interprets data differently.
Inconsistent colors, scales, layouts across views.

Governance Gaps

No ownership model for metrics or visuals.
Versioning and access are afterthoughts.

THE STANDARD APPROACH

The 3-Layer Design System

This model works well for consumer apps and typical product user interface.

01

Primitives

Design tokens, colors, spacing, typography, iconography

02

Components

Buttons, inputs, cards, modals, navigation elements

03

Patterns / Semantics

Page layouts, interaction flows, composition rules

Clean. Predictable. Familiar. But enterprise analytics demands more.

THE GAP

Why Three Layers Are Not Enough

Enterprise analytics is complex, data-driven, and scale-sensitive.

Tables Break at Scale

Standard tables choke on 1M+ rows. No virtualization, no streaming.

Visuals get unreadable

Clean at 100 points, noise at 100K. No rules for aggregation or sampling.

Scalability is the core gap

Built for visual polish, not for scale, latency, or query cost.

Nothing drop-in-and-go

Teams stitch libraries by hand. Nothing you can drop in and trust.

We know why it breaks.

Now let's talk about how to fix it.

THE PROPOSAL

A 7-Layer Model for Analytics Design Systems

Each layer builds on the one below. Each has a clear purpose.

7	**Governance**	Ownership, versioning, access, change management
6	**Workflow Patterns**	Drill-down, cross-filter, explore-to-dashboard flows
5	**Semantics / Interpretation**	Metric definitions, units, thresholds, meaning
4	**Visualization Primitives**	Axes, scales, encodings, chart building blocks
3	**Data Components**	Tables, filters, query builders, data-aware inputs
2	**UI Components**	Buttons, inputs, layout, navigation
1	**Design Token Foundation**	Colors, spacing, type, data-viz tokens

LAYER 1 OF 7

Design Token Foundation

1

The atomic layer - values that everything else references.

Data-viz color palettes	Sequential, diverging, and categorical scales purpose-built for analytics.
Density modes	Dashboards need tighter spacing than general apps. Tokens adapt per context.
Semantic tokens	Success, warning, threshold, anomaly - meaning baked into the token layer.
Platform-agnostic output	Tokens compile to CSS, iOS, Android, and Figma from a single source.

LAYER 2 OF 7

UI Components

2

The familiar library - still necessary, but not sufficient alone.

Standard controls	Buttons, inputs, selects, modals, tabs, breadcrumbs, layout primitives.
Built on tokens	Every component consumes tokens. No hardcoded values.
Unaware of data	These components have no opinion about rows, metrics, or queries.
Where most design systems stop	For consumer products this is enough. For analytics, this is just the start.

LAYER 3 OF 7

Data Components

3

UI that knows it is handling data at scale.

Virtualized tables	Handle millions of rows through virtualization, streaming, and lazy loading.
Query builders and filter panels	Data-aware inputs: metric pickers, dimension selectors, faceted search.
Stateful by design	Loading states, empty states, partial-data states, error boundaries.
Column configuration and saved views	Users personalize without breaking shared definitions.

LAYER 4 OF 7

Visualization Primitives

4

The building blocks of every chart - not the charts themselves.

Axes, scales, legends, grids, tooltips	Composable pieces that assemble into any chart type.
Visual encodings	Position, color, size, shape - with accessibility built in.
Aggregation and sampling	Rules for what to show when data exceeds pixel density.
Interaction primitives	Zoom, brush, pan, focus+context, linked highlighting.

LAYER 5 OF 7

Semantics / Interpretation

5

What the data actually means - the layer most design systems ignore entirely.

Metric definitions	One source of truth for 'revenue', 'churn', 'latency'. No more five definitions.
Units, formatting, precision	Consistent display rules: currency, percentages, timestamps, localization.
Thresholds and benchmarks	Targets, goals, and anomaly rules embedded in the system.
Shared vocabulary	A common language between design, data engineering, and product.

LAYER 6 OF 7

Workflow Patterns

6

How users actually explore and interact with analytics.

Drill-down and drill-across	Navigating from summary to detail, from one dimension to another.
Cross-filtering	Selecting in one view filters all linked views automatically.
Explore-to-dashboard	Saving ad hoc analysis as a persistent, shared dashboard.
Comparative views	Period-over-period, segment-over-segment, baseline comparison.

LAYER 7 OF 7

Governance

7

The layer that keeps the system trustworthy at organizational scale.

Ownership Model

Every metric, component, and pattern has a named owner. Accountability prevents drift.

Versioning Discipline

Changes to a metric are changes to history. Semantic versioning with migration paths.

Access & Sensitivity

Data classification and role-based access baked into the design system, not bolted on.

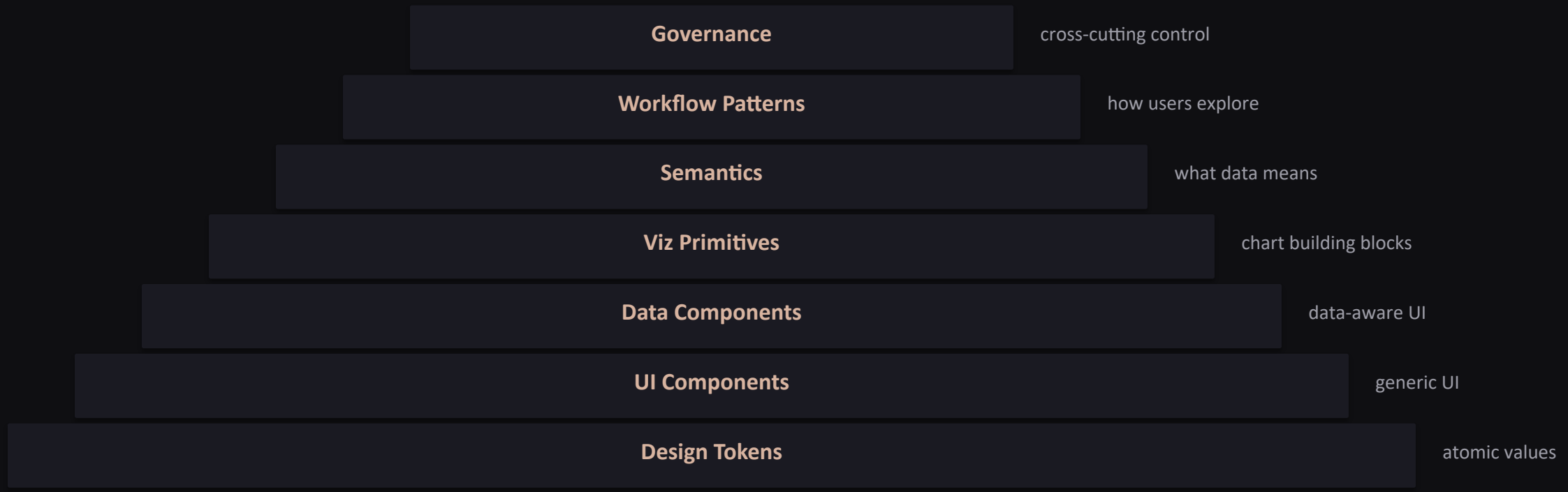
Contribution Workflow

Review process for new components and metrics. Quality gates and deprecation policies.

ARCHITECTURE VIEW

How the Seven Layers Interconnect

Lower layers are foundations. Higher layers add meaning and control.



GETTING STARTED

A Phased Approach to Adoption

You don't build all seven layers on day one. Start where the pain is.

Foundation	Establish design tokens and governance first. Cheapest layers, highest leverage.
Data Scale	Invest in data components and viz primitives. This is where users feel the pain most.
Semantics	Unify metric definitions across teams. Organizational work, not just technical.
Patterns	Codify workflow patterns once you see them repeat. Emerge from use, don't prescribe upfront.

TAKEAWAYS

What to Take Back to Your Team

01 Three layers are not enough for analytics.

02 Seven layers give you the full picture.

03 Governance is not a nice-to-have.

04 Start at the edges, work inwards.

CLOSING

Build Systems That Scale With Your Organization

Pick one painful layer and fix it. Success compounds from there.

Track reuse rates, consistency scores, and time-to-dashboard.

A design system without governance is just a component library.

THANK YOU

Sonali Priya

hello@sonalipriya.com

linkedin.com/in/priyasonali

I'd love to hear how you're thinking about this in your own platforms.