Composable CDP or Hybrid CDP?

Choosing the Right Architecture for Real-Time Personalization

> Ten years of booking history might tell an airline that someone usually travels for business, but the last three minutes on the app shows they're desperately trying to add family baggage to a flight departing in two hours. An AI agent or service rep can't solve that crisis with a static warehouse profile that just flipped a single flight_search flag. They need the full streaming profile - live cart additions, countdown tickers, and session state recomputed on the fly. Real relevance comes from event-time context, not decade-old habits.

Mike Anderson, Founder & CTO, Tealium

Foreword

As more customer data moves into cloud warehouses, teams are asking this question: when is a composable, warehouse-paced CDP enough, and when does every click need to matter in real time?

This ebook cuts through that confusion by showing how data freshness shapes personalization, AI, and service experiences.

It explains where composable CDPs shine, where they fall short, and how a hybrid architecture lets you keep your warehouse at the center while adding a real-time layer for truly event-time decisions.

Executive Summary

Is your API returning a whole updated profile, or just a single refreshed audience flag attached to stale warehouse attributes?

COMPOSABLE / SAME-SESSION

```
{
  "audiences": [
    "comparison_tool_engager", // FRESH — flipped by the same-session event
    "high_value",              // STALE — last warehouse cycle
    "mobile_user"              // STALE — last warehouse cycle
  ],
  "synced_attributes": {
    "ltv_score": 4280,                  // STALE — last warehouse cycle
    "churn_propensity": 0.72,           // STALE — last warehouse cycle
    "predicted_upgrade_propensity": 0.61, // STALE — last warehouse cycle
    "current_handset_age_months": 28    // STALE — last warehouse cycle
  }
}
```

A *simplified same-session composable payload* may look like this.

REAL-TIME STREAMING

```
{
  "visitor_id": "abc123",
  "audiences": [
    "VIP",                    // LIVE — re-evaluated on every event
    "Cart Abandoner - last 24h", // LIVE
    "Repeat Buyer",           // LIVE
    "Comparison Tool Engager"   // LIVE
  ],
  "attributes": {
    "ltv_score": 4280,      // LIVE — refreshed with the latest profile evaluation
    "churn_propensity": 0.72,          // LIVE
    "predicted_upgrade_propensity": 0.61, // LIVE
    "current_handset_age_months": 28,    // LIVE
    "last_product_viewed": "iphone_17_pro", // LIVE — this session, this second
    "pages_in_session": 7,             // LIVE — counter updates on each event
    "cart_value_now": 1280,            // LIVE — includes the latest basket event
    "seconds_since_last_event": 3,     // LIVE — derived at response time
    "session_intent_score": 0.84       // LIVE — recalculated from current behavior
  }
}
```

A *real-time streaming profile payload* may look like this.

In real-time experiences, the critical question isn't just how fast your API responds, but what it's actually returning. Many composable, warehouse-paced models serve a pre-computed snapshot from the last warehouse run, with a thin same-session layer that only re-evaluates one audience tied to the trigger. By contrast, a streaming real-time profile recomputes the entire state—attributes, scores, audiences, counters, and session signals like pages_in_session or cart_value_now—on every qualifying event before the next API call.

This ebook will help you test your current stack against that standard: is your API returning a whole updated profile, or just a single refreshed audience flag attached to stale warehouse attributes?

Why buying teams get stuck at the intersection of warehouse and real-time

Enterprises, like yours, that have decided to centralize their customer 360 or single customer view in a cloud data warehouse such as Snowflake, Databricks, or BigQuery increasingly face a practical architecture choice.

Should you go with a composable CDP, built around warehouse-resident profiles and activation, or a data orchestration platform that spans real-time, cloud-first, and hybrid architectures that can operate with warehouse-resident data while also supporting true real-time decisioning when your use case demands it?

This is where many buying teams get stuck. Identity resolution, data modeling, and audience creation are usually well understood. The nuance appears when the conversation moves from segmentation to real-time personalization. It is not only important to ask which platform is faster, but what sits behind the API, how fresh the underlying profile really is, and whether today's architecture will still work for your tomorrow's use cases.

The most useful way to think about this is simple: the warehouse remains an excellent place to persist history, compute models, and maintain a durable system of record, while a real-time layer becomes important when the next customer interaction depends on what happened seconds ago, not what was true at the last warehouse refresh. Tealium's current architecture is designed to support both patterns from one operational surface: Tealium Data Cloud Activation paired with Context API for warehouse-paced use cases, and Tealium CDP paired with Context API for use cases that need per-event freshness.

- Tealium Data Cloud Activation is Tealium's cloud-native activation layer for modeled customer data already in the warehouse, designed to activate that data across channels, systems, and real-time experiences without creating a second source of truth.
- Tealium CDP is Tealium's real-time customer data platform.
- Context API is Tealium's governed, low-latency context layer for AI agents, apps, and real-time experiences.

Matching data freshness to the use case

A lot of the market debate we see out there is framed as warehouse-native versus traditional Customer Data Platform (CDP). In practice, that setup oversimplifies how enterprises actually operate. Many of them are not looking to reject the warehouse; they are looking to decide whether the warehouse alone is enough for every activation and personalization moment.

For some use cases, a warehouse-paced profile is absolutely the right answer. If a brand wants to personalize a homepage banner using a propensity score refreshed every 30 minutes, enrich outbound audiences before an email send, or expose household-level attributes in an account portal, then a composable model can work well.

Tealium explicitly supports that model too through Tealium Data Cloud Activation + Context API, where profile retrieval is real-time but profile freshness follows the warehouse cadence the customer chooses. That matters because the decision does not need to be binary. Enterprises do not have to choose between warehouse-centric and real-time as if they are mutually exclusive architectural camps. They can choose the right freshness model per use case.

Where composable CDPs are a strong fit

A composable CDP approach is often a sensible fit when the experience does not depend on within-session behavioral change. In these scenarios, the profile is still valuable, but the business outcome is not harmed if it reflects the last warehouse update rather than the last click, event, or interaction.

Typical examples include:

- Warehouse-computed propensity or eligibility scores used to personalize content slots or product recommendations.
- Pre-send enrichment for outbound channels such as email or SMS, where the campaign workflow can tolerate warehouse-paced freshness.
- Self-service portals or account management experiences where long-term profile state, household relationships, or account history matter more than live session behavior.
- Internal AI or analytics use cases operating against features that are already modeled in the warehouse.

In other words, if the decision window is measured in hours, a warehouse-first composable pattern can be entirely appropriate. That is precisely why this category has gained traction.

Where true real-time becomes decisive

The architecture changes when the customer experience depends on the current session state. That is the point at which a buyer needs to look beyond audience membership and ask whether the underlying profile is recomputed on each event, or whether it is still fundamentally a warehouse snapshot with some faster-triggered flags around it.

This is especially important in use cases such as:

- **In-session web or app personalization,** where the next page, component, or offer should reflect the customer's last few clicks.
- **Anonymous first-page personalization,** where there is no existing warehouse record yet, but intent signals are already available from the first event.
- **Unknown-to-known identity stitching mid-session,** where anonymous behavior needs to be merged into a known profile immediately after login or form submission.
- **Cart recovery or suppression** inside the active session, where reacting after the warehouse refresh is simply too late.

This is the practical dividing line. If the experience depends on what the customer did this session, then the organization needs more than low-latency retrieval. It needs a profile that is actually fresh to the event.

Why two APIs can look similar but behave differently

This is where confusion often creeps in for buyers. On paper, two APIs can appear similar because both return profile data quickly, both expose audiences, and both can surface attributes.

But the important distinction is architectural. In many composable models from vendors today, the personalization API reads from a pre-computed cache of warehouse-derived audience assignments, then relies on a thin same-session layer to re-evaluate audience membership when qualifying real-time events arrive

In the Tealium CDP model, Context API reads from a continuously updated streaming profile where every qualifying event can update the whole profile state.

That difference matters because there is a big gap between:

- A profile where one audience flag changes quickly, and
- A profile where attributes, scores, audiences, counters, and identity state are all current to the last event.

For a marketer running batch activation, that distinction may not matter much. For a chatbot, personalization engine, or call-center application trying to decide what to do next in the moment, it matters a great deal.

Payload comparison: similar shape, very different freshness

One reason this category can be confusing is that the payloads can look superficially similar. Both can return audiences and attributes. The difference is in what was actually refreshed when the event happened.

```JSON
{
  "audiences": [
    "comparison_tool_engager", // FRESH — flipped by the same-session trigger event
    "high_value",              // STALE — last warehouse cycle
    "mobile_user"              // STALE — last warehouse cycle
  ],
  "synced_attributes": {
    "ltv_score": 4280,                  // STALE — last warehouse cycle
    "churn_propensity": 0.72,           // STALE — last warehouse cycle
    "predicted_upgrade_propensity": 0.61, // STALE — last warehouse cycle
    "current_handset_age_months": 28    // STALE — last warehouse cycle
  }
}
```

A *simplified same-session composable payload* may look like this:

```JSON
{
  "visitor_id": "abc123",
  "audiences": [
    "VIP",                    // LIVE — re-evaluated on every event
    "Cart Abandoner - last 24h", // LIVE
    "Repeat Buyer",           // LIVE
    "Comparison Tool Engager"   // LIVE
  ],
  "attributes": {
    "ltv_score": 4280,     // LIVE — refreshed with the latest profile evaluation
    "churn_propensity": 0.72,           // LIVE
    "predicted_upgrade_propensity": 0.61, // LIVE
    "current_handset_age_months": 28,    // LIVE
    "last_product_viewed": "iphone_17_pro", // LIVE — this session, this second
    "pages_in_session": 7,           // LIVE — counter updates on each event
    "cart_value_now": 1280,          // LIVE — includes the latest basket event
    "seconds_since_last_event": 3,    // LIVE — derived at response time
    "session_intent_score": 0.84      // LIVE — recalculated from current behavior
  }
}
```

A *real-time streaming profile payload* may look like this:

At a glance, both look usable. Both expose segments of customer context. But the freshness model underneath them is very different. What changes when the event fires?

In a same-session composable pattern, the event typically re-evaluates audience membership for the audience tied to that trigger. That means one boolean or cohort assignment may be fresh, while many of the surrounding synced attributes still reflect the last warehouse update.

In a streaming real-time profile, the event updates the whole profile state. That means the audiences, counters, derived session attributes, and event-driven scores are recalculated together before the next API response is returned.

Why attribute freshness matters

This difference becomes important when a buyer looks beyond whether the field exists and asks what point in time it represents.

For example:

- ltv_score may exist in both payloads, but in a warehouse-paced model it may reflect the last warehouse run, whereas in a streaming model it can be refreshed as part of the latest profile evaluation.
- predicted_upgrade_propensity may look identical in name, but one version may represent the last modeled snapshot while
- the other reflects the current customer state including the most recent event.
- pages_in_session, cart_value_now, or seconds_since_last_event are fundamentally different. These are not just attributes that need to be retrieved quickly; they need to be derived from live session behavior.
- Session-level intent is often the most important signal in a real-time experience, but it is also the easiest to lose when the profile is only partially refreshed.

The practical takeaway is that buyers should not only ask, "Can this attribute be returned?" They should ask, "When does this attribute get recalculated, and does it reflect the event that just happened?" That is the real test of whether the architecture is simply low-latency, or truly real-time.

How Spark NZ used this exact whole-profile freshness to deliver 150M AI offers a month

Spark New Zealand feeds live behavioral data into an AI engine in <300ms. An AI recommendation model cannot calculate a Next-Best-Offer from an isolated trigger, it requires the entire recalculated profile vector (recent views, category affinity counters, current cart state).

+22% Next Best Offer Conversion

Why Warehouse-Only Fails: Feeding an in-session AI engine requires continuous re-evaluation of the entire customer feature set in <300ms, not just batch audience syncs.

Discover the full story

The future use cases should influence today's architecture decision

Many buying teams evaluate against today's requirements only. That is understandable, but risky. The next generation of customer engagement will increasingly depend on chat, call-center augmentation, and agentic commerce. Autonomous AI agents and MCP servers require a unified Context API surface that serves both Cloud Data Activation (warehouse traits) and Real-Time CDP (live telemetry) in a single call. Those use cases may not all be on the phase-one list, but they are becoming strategically important enough that the chosen architecture should not make them difficult or impossible later.

1. Chat and AI assistants

A chatbot or AI assistant becomes materially more useful when it can access live customer context in the middle of a conversation and when the outcome of that interaction can enrich the profile for the next turn. In practice, this is where an architecture with a closed-loop pattern of event → profile update → API retrieval → next interaction becomes materially stronger.

2. Call-center and service handoff

The call-center scenario is one of the clearest real-time tests. If a customer was browsing products, completing a quote, or showing signs of friction a few minutes before they called, the service agent should ideally have that context at the moment of connection, not after the next warehouse cycle. This is exactly the kind of use case where phone-number-based lookup, sub-second retrieval, and current session context become highly valuable.

3. Agentic commerce

Agentic commerce introduces an even newer challenge: the customer may purchase through an AI agent or assistant without ever visiting the brand's site in the traditional way. That creates a blind spot for architectures that depend on conventional web-session collection and delayed warehouse processing. It is a future-facing requirement, but one that is becoming increasingly relevant for maintaining an accurate and usable customer view.

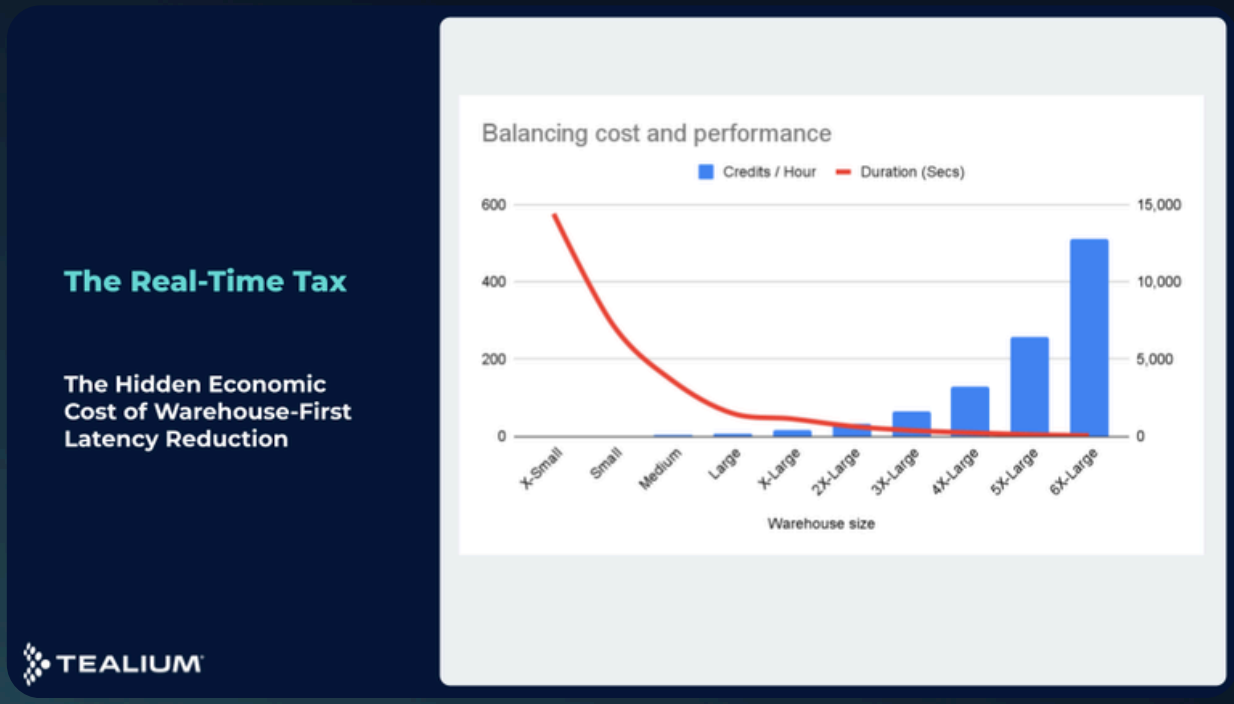
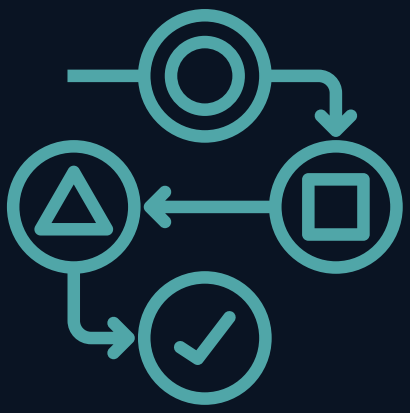


Figure 1: Warehouse compute cost can rise non-linearly as teams try to reduce query duration toward real-time, which is why freshness requirements and economics need to be evaluated together.

A practical framework for buyers



For most enterprises, the right answer is not composable is good or hybrid is good.

The right answer is:

- Use a warehouse-paced composable approach when the use case depends mainly on durable history, modeled attributes, and activation windows measured in minutes or hours.
- Use a real-time streaming profile when the next decision depends on what happened moments ago, especially for anonymous users, active sessions, service interactions, or live AI experiences.
- Prefer a hybrid architecture when you want to preserve warehouse investments without ruling out future use cases that require genuine event-time freshness.

That is the nuance buyers should focus on. The question is not whether a warehouse-centered architecture is valid. It clearly is. The question is whether the architecture chosen today leaves room for the customer experiences the business is likely to need next.

In that sense, the strongest long-term design principle may be this: let the warehouse remain the system of record, but do not assume it must also be the only system of action.

Legal & General is unifying real-time data & lifetime value with Tealium & Snowflake

Legal & General
When an applicant struggles on a financial form and is routed to a phone rep, the agent needs full live session context (fields changed, time on page, income values) plus the warehouse LTV score. A composable same-session trigger would only send a single flag (application_struggled = true), without the rich session details the agent needs.

+54% Call-to-Lead Conversion

Why Warehouse-Only Fails: Call center agents need the full live clickstream and form telemetry at the moment of connection, not an isolated trigger flag hours before the next warehouse sync.

Discover the full story

I truly believe that insights are meaningless if you have no capacity or no capability to drive change with the insight that you gathered. And that is where some organizations can fall down. Insights never leave a powerpoint slide

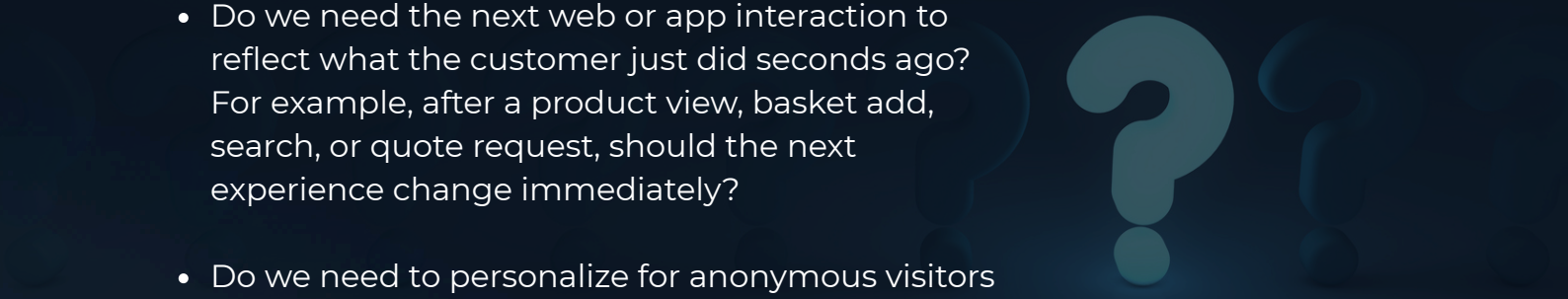
Joshua Williams, Senior Marketing Technology Engineer, Legal & General

Watch the full 26-mins session.

Questions buyers should ask themselves

Before choosing an architecture, buyers should ask which of their current or likely future use cases require the profile to be re-evaluated within the same session after an event occurs.

Useful questions include:



- Do we need the next web or app interaction to reflect what the customer just did seconds ago? For example, after a product view, basket add, search, or quote request, should the next experience change immediately?
- Do we need to personalize for anonymous visitors before they are known in the warehouse? If yes, first-page and in-session personalization become real-time profile problems rather than warehouse refresh problems.
- Do we need unknown-to-known stitching to happen during the live session? If a user logs in, submits a form, or identifies mid-journey, should the prior anonymous behavior immediately influence the next interaction?
- Do we need suppression or recovery logic to respond within the session? For example, should a cart recovery message, offer suppression, or next-best action update as soon as a purchase or abandonment event happens?

Questions buyers should ask themselves

- Do we need chat or AI assistants to react to live behavioral context? If the assistant's next response should reflect what happened in the previous turn or the previous click, the underlying profile has to update at event time.
- Do we need a call-center agent to see digital behavior from the last few minutes at the moment a call connects? If yes, the profile must be refreshed before the next interaction, not at the next warehouse cycle.
- Do we expect agentic commerce or agent-mediated interactions to become important? If a customer acts through an AI assistant rather than a website session, how quickly should that interaction be reflected in the profile used elsewhere?
- Can our architecture distinguish between use cases that are warehouse-paced and use cases that are truly event-driven? If both exist, a hybrid approach may be the most resilient design.

The key evaluation criterion is simple: if a use case depends on the customer's profile being materially different because of an event that happened moments ago, then the buyer should treat that as a genuine real-time requirement and test the architecture against that standard.

Talk to Tealium about matching data freshness to your use cases.

Book a meeting →

See how Tealium's hybrid CDP powers both warehouse-paced and real-time experiences.

See how Tealium helps →

Further resources:

Read the blog (5 mins) →

Listen to this conversation →

More customer story shorts →

TEALIUM

About Tealium

Tealium delivers trusted data for AI at enterprise scale through its customer data orchestration platform. As a foundational data layer, it powers a hybrid CDP with real-time streaming, context, and 1,300+ integrations. Tealium enables enriched, consented data to accelerate AI, improve efficiency, and power meaningful customer experiences worldwide.

tealium.com
