

When two systems tell the same customer different things

The email says the plan was changed, and the account page says it was not. Both systems are working, and the customer is the one who finds out they disagree.

Author	Amit Govil, Founder and CEO, XONIK
Published	October 22, 2026
Reading time	8 min read
Online	https://xonik.com/insights/perspectives/when-two-systems-tell-the-same-customer-different-things/

IN BRIEF

A customer hears two answers when two systems each hold a copy of the same fact and change it on their own trigger, at their own time. Staff then believe whichever screen is open. Naming one system as the source for each fact, showing where an answer came from and comparing the copies on a schedule narrows the gap.

KEY FINDINGS

1. A copy fed by events or syncs is correct eventually, and eventually is not a time a customer can see.

Events can arrive out of order, be retried for a limited period and arrive more than once, and delivery cannot always be relied on. Nothing is broken, and no system is answerable for the interval.

2. The same word can mean different things in two systems, and the mapping between them is rarely written down.

Active in billing means the invoice is paid; active in support means the account is open. A sync that maps one onto the other makes a choice nobody recorded.

3. Staff invent rules for which screen to believe, and they repair copies instead of the source.

The rules pass from person to person and are learned by being wrong in front of a customer. A correction typed into a copy is overwritten by the next sync, so the customer is told twice that it is fixed and watches it revert.

4. Every owner's dashboard is correct, so nobody owns the difference.

The disagreement exists only between systems, where no dashboard looks. A lag is acceptable only if somebody chose it, and in an estate that has grown by adding tools nobody did.

5. Naming one system as the source for each fact needs no new platform to begin.

A register of the fact, its source, its readers and the allowed staleness is enough, with messages built at sending time and copies compared on a schedule. It has worked when the person answering knows which system to believe and can say why.

How do two systems come to give one customer two answers?

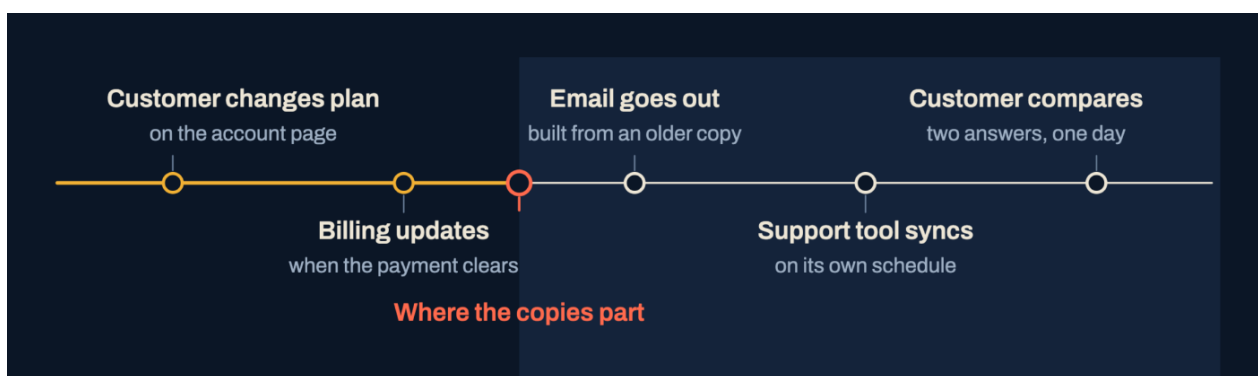
A customer's plan, address or status is rarely stored once. Billing needs it to charge, the messaging tool to choose a template, the support tool to greet the caller and the account page to show it. Each system keeps its own copy, and each copy is changed by a different trigger: billing when a payment clears, messaging when a campaign is built, support when an agent saves a note. None of them is broken. The disagreement lives in the interval between the triggers, and in that interval no system is answerable for it. Exhibit 1 follows one plan change through the systems that hold it.

These things widen the interval. One is the route a change takes from one system to another. Event feeds and scheduled syncs are both common, and neither delivers at the instant of the change. Stripe's webhook documentation states that events may arrive in a different order from the one in which they were generated, that Stripe keeps retrying a failed delivery for a limited period, and that a receiver should log event identifiers because the same event can arrive more than once. Shopify's webhook guidance says the same about ordering and adds that delivery cannot always be relied on. A copy fed this way is correct eventually, and eventually is not a time a customer can see.

Another is meaning. A status of active in billing means the invoice is paid, and active in support means the account is open. A sync that maps one onto the other makes a choice, and the choice is rarely written down. The last is the passage of time itself: a copy that was right when it was taken is wrong an hour later and looks exactly as right as before.

EXHIBIT 1

One plan change, four systems that part company



The first events are one change. Everything after the mark is each system catching up in its own time, and the customer is the only observer of all of them.

What does the customer actually experience?

Take a guest who moves an arrival date. The booking site accepts the change and shows the new date. The confirmation email, assembled by a messaging tool from an earlier copy of the reservation, still carries the

old one. The property's own system receives the change on its next sync, so for a stretch of hours the front desk holds the old date as well. The guest now has two written statements from one company and no way to tell which will be honored. They telephone, and the agent opens whichever screen is nearest. If that is the stale one, the agent confirms the wrong date with complete confidence, and the error is now spoken, which is harder to retract than a screen.

The customer does not experience a synchronization delay. They experience a company that does not know its own mind about their booking. The cost is not the lost minute. Every later message from the firm is read with doubt, including the ones that are correct.

What do staff do when two records disagree?

They invent rules. An agent learns that billing is right about money, the customer record is right about contact details, and the support tool runs behind on certain days. Nobody writes this down. It passes from one person to the next, and new staff learn it by being wrong in front of a customer. Two agents on one team can hold different rules and give different answers to the same question.

The other habit is repair in place. The agent sees a wrong address on the screen, corrects it and tells the customer it is fixed. The system that owns the address still holds the old one, and its next sync overwrites the correction. The customer is told twice that it is fixed and watches it revert. The correction was made to a copy, and a copy is not where a fact lives.

Both habits are sensible for the person using them. Together they turn a data problem into a training problem and a trust problem, and they bury the evidence. If every disagreement is quietly absorbed at the desk, nobody measures how often the copies part, and nobody is asked to close the gap.

| The correction was made to a copy, and a copy is not where a fact lives.

Why does nobody own the difference?

Each system has an owner who watches that system, and each owner's dashboard is correct. The disagreement exists only between systems, where no dashboard looks. The one person who sees both copies side by side is the customer.

The UK Government Data Quality Framework, published by the Cabinet Office on 3 December 2020, defines two of its quality dimensions in terms that fit. Consistency is the degree to which values do not contradict other values representing the same entity, including values in another data set. Timeliness asks whether the lag between a change and its availability suits the intended use, and the framework's own example fixes an agreed limit for how long a changed contact detail may take to reach the database, treating a longer wait as a breach of the rule. The lesson for a business is that a lag is acceptable only if somebody chose it. In an estate that has grown by adding tools, nobody did. The lag is whatever the sync happens to produce.

The framework also observes that where quality problems are found, the symptoms are often treated instead of the cause. A correction typed into the screen in front of an agent is exactly that: a symptom treated, at a copy, while the source goes unchanged.

What makes one system answer for each fact?

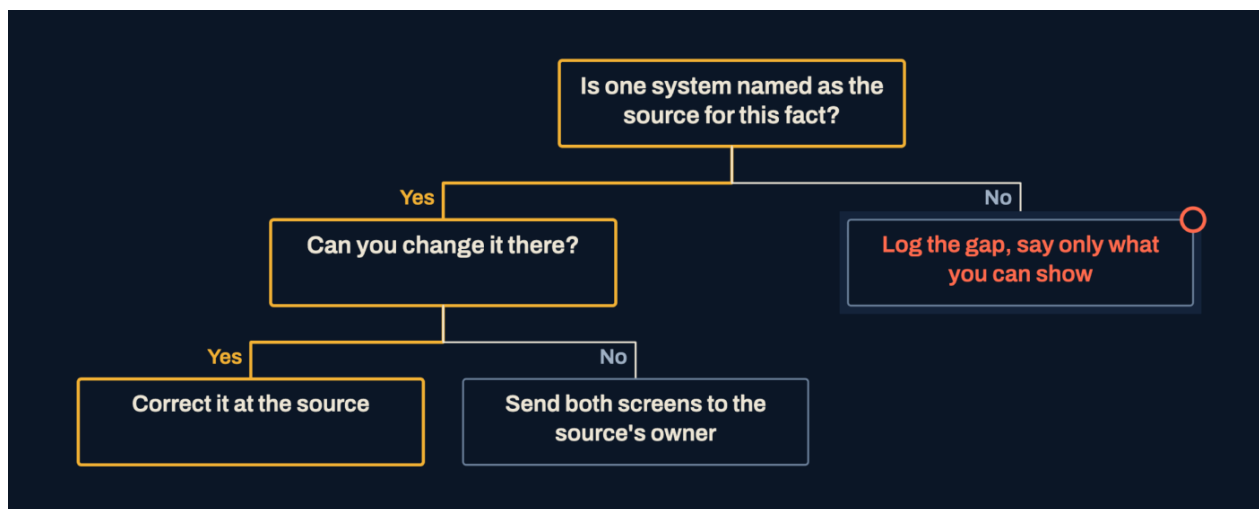
A decision, fact by fact. The source for a fact is the system where it is created or changed, and every other system either reads from it or says on its screen that it holds a copy. A short register is enough to begin: the fact, its source, the systems that read it, and how stale a reader is allowed to be, stated in the unit the business cares about, whether minutes or days.

These working rules follow. Messages to customers are built at the moment of sending, by reading the source, and not from a copy stored when the campaign was assembled. Screens that show a copy say where it came from and when it was last confirmed, so that a stale value looks stale. Corrections are made at the source and nowhere else, which means every customer-facing person needs a route to the source's owner and not merely to a form. And the copies are compared on a schedule, record by record, for the facts that customers hear differently, with every difference sent to the source's owner. This needs no new platform to begin. A query in each system and a script that matches on identifier will show where the work is.

Exhibit 2 shows the decision a customer-facing person should be able to make in seconds once a register exists, and what happens when it does not.

EXHIBIT 2

What to do when two screens disagree



The tree is short because the register has already answered the hard question. Without a named source the only sound outcome is to record the gap.

How do you know it has worked?

The test is not that the systems never differ. They will, for the length of the interval. The test is that when they differ, the person answering the customer knows which system to believe and can say why, the customer hears one answer, and the difference is recorded where someone can close it. Ask people who answer customers to name the screen they trust for a plan, an address and a status. If they name the same screen, the register is working. If they name different ones, you have found the first fact to settle.

Sources read for this piece

Every passage relied on was read in the original on 20 October 2026. What staff do, and how copies part, comes from our delivery work and is set out as a pattern; no figure supports it.

1. [Stripe, Receive Stripe events in your webhook endpoint: events out of order, repeated deliveries, retries](#)
2. [Shopify, About webhooks: ordering and delivery not assured](#)
3. [Cabinet Office, The Government Data Quality Framework, 3 December 2020: definitions of consistency and timeliness, and treating issues at source](#)

USE THIS

[Read why systems and data stop agreeing](#)

[Use the architecture review checklist](#)

Ask the people who answer customers which two screens they open side by side. The pair they name is your first reconciliation.

This document was made from the article at

<https://xonik.com/insights/perspectives/when-two-systems-tell-the-same-customer-different-things/>. The online article is the current version.