Habit trackers watch you fail. FlexMax figures out why.

The behavioral engine — how it captures, how it reasons, and what it produces.

Download as PDF

FlexMax · Behavioral Engine
Technical & product overview

September 2026

THE PROBLEM

Time isn't traded away. It evaporates.

There is no moment of choosing distraction over the plan. There is only a later moment of noticing the plan didn't happen.

Most people who fail a schedule don't lack motivation and don't lack a plan. They produce an accurate, realistic day — and then fail to execute it, repeatedly, without ever making a conscious decision to abandon it.

Then the gap between plan and reality grows large enough that reviewing it becomes aversive. Re-engaging would mean confronting the whole accumulated shortfall at once. So it gets avoided entirely.

That's the uninstall moment — for every planner, tracker, and calendar that person has ever tried.

Calendars

The schedule was accurate. It was ignored. A calendar entry is a suggestion with no weight behind it.

Habit trackers

Rendered the shortfall as a verdict — which accelerates the freeze rather than relieving it.

All of them

Nothing distinguished a deliberate rest from a silent collapse. Both showed up as the same empty row.

THE DIFFERENCE

Every other app records the plan. This one records what actually happened.

WHAT A PLANNER STORES

Gym, 7:00 AM — not done.

A record of intent, and a record of its absence. Nothing about why. Miss it forty times and the app schedules it a forty-first time, identical.

WHAT FLEXMAX STORES

Gym, 7:00 AM — missed, acknowledged at 9:14 AM.
Reason: ran out of time.
Written: "I let my deep work spill into this."
Rescheduled 13 times this month, always later.
Preceded by a night that ended after midnight.

A record of the mechanism. That's what can be reasoned about.

The distinction that makes it work: **missed** means the user engaged and admitted it. **Unaccounted** means silence.

The same completion rate describes both. They are not the same behavior. One is accountability; the other is the drift itself. An engine that can't tell them apart is measuring the wrong thing.

ARCHITECTURE

SQL computes. The model narrates. Nothing learns in the dark.

Every number — counts, rates, trends, correlations — is computed in the database, where arithmetic is exact and auditable. The language model receives only finished figures and the user's own words. Its single job is to name what's left and cite the evidence. It never calculates, never invents a statistic, and never decides strategy.

LAYER 01 · CAPTURE

Record what happens, including the silence

Check-ins, completion quality, miss reasons, reschedule history, real bedtimes, nudge responses. Data not captured today cannot be recovered later.

↓

LAYER 02 · COMPUTE

One SQL function turns raw rows into finished facts

Per-block records, day shapes, weekly trends, reschedule drift, quality trajectories, ordered within-day pairs — plus the integrity filters that decide what may be concluded at all.

↓

LAYER 03 · NARRATE

One AI call per user, twice a week

The model reads the finished numbers and the user's verbatim reflections, then writes up to five ranked insights with the evidence attached — kinds `structural`, `causal`, `pattern`, and `strength`. Results are stored, not regenerated.

↓

LAYER 04 · SURFACE

Injected where it's useful — free, no further AI

The morning card. The recovery sheet when something breaks. The weekly review. Stored beliefs are read, not recomputed.

LAYER 01 · CAPTURE

The signals

Most are invisible to the user. All of them are things a planner throws away.

OUTCOME

What happened, and whether you said so

QUALITY

How well it went, not just whether

Crushed it, partial, pulled away. A block still completing but degrading is an earlier warning

Completed, missed, or unaccounted — with a timestamp for when it was acknowledged, so recovery speed is measurable.

than a miss.

REFLECTION

In the user's own words

Free text on why, and what they'd change. The highest-signal column in the system — people narrate their own patterns better than any model invents them.

MOVEMENT

Every reschedule, permanently

Which blocks get pushed later, how far, how often. Overwriting a time destroys the strongest available signal, so nothing is overwritten.

BOUNDARIES

When the day actually started and ended

Real sleep and wake times, captured retroactively in the morning. Sleep is a moment you cross, not a block you complete.

INTERVENTION

Whether the nudge worked

Every cutoff notification, the response chosen, and the block's eventual outcome. This is the record that compounds.

LAYER 02 · COMPUTE

Behavioral data lies. Every filter was earned.

Each of these exists for a reason: a simpler version produced a confident, false conclusion. They ship inside the data payload so the narrator cannot drift beyond what the evidence supports.

Scheduled times are not behavioral records

A recurring block reports its template time every day regardless of what happened. An early design correlated "late nights" against it and found the identical value on all forty rows — it was reading the plan,

not the behavior. The engine may never claim a block "ran until" a time.

Untracked blocks are excluded entirely

A block qualifies only with at least three check-ins and engagement on a quarter of its instances. A single accidental completion once ranked "Sleep" as the user's largest failure at 24 of 26 — nobody checks off sleep.

Days the app wasn't used don't count

Instances generate on demand, so planning ahead creates rows nobody touched. Read naively, this suggested the user abandoned three quarters of their commitments — false, and precisely the shame signal the product exists to avoid.

Never trend from a partial period

An early draft claimed a four-day winning streak from morning blocks alone. Whole-day data showed two of those four were near-total collapses, and the most recent complete week was the worst of the month. Today is excluded; partial weeks are omitted.

One block at a time isn't enough. The day is a sequence.

Per-block rates answer *how often*. They cannot answer *when*. A block that fails 40% of the time might fail at random — or it might fail on exactly the days something earlier went wrong, which is a different problem with a different fix.

So the engine looks at ordered pairs within a day: for every block that ends before another starts, does the first one's outcome predict the second's?

Both directions are kept. An earlier block **failing** and taking a later one with it is one relationship. An earlier block **succeeding** and consuming the later one's time is the opposite. Both are real. An engine that only looks for one will find it whether or not it's there — and on this dataset, every pattern that survived pointed the way the original hypothesis didn't.

Persistence across two windows

With nine blocks there are up to 72 ordered pairs. Report the largest gap from 72 comparisons and you will find something in pure noise. A pattern is only reported if it holds in the last thirty days *and* the thirty before — same direction, both times. A real dependency survives that. A coincidence does not.

The day-level control

Two blocks failing together might mean they are linked, or it might just mean it was a bad day. So the engine measures what the rest of that day did. If the whole day moves as much as the pair does, the pair is not the finding. One candidate here had a 41-point gap and was dropped — 39 of those points were explained by the day going badly overall.

Silence is not refusal

A later block failing mostly as *unaccounted* may mean the user stopped logging, not that they gave up. Pairs where the failures are mostly silence are excluded. The distinction between missed and unaccounted runs through the whole system; it matters most here.

Across one month of real use, the engine found **five candidate patterns and reported one.** Two failed persistence. One was explained by the day. One was too thin on one side to mean anything. Every other system in this category is built to find you more.

LAYER 03 · NARRATE — WHAT IT TESTS

It can't tell you why.
It can tell you what it isn't.

A correlation is not an explanation. When the engine finds that two blocks move together, there are three candidate reasons, and they are fixed — the model cannot invent a fourth:

Carry — the earlier block's outcome carries into the later one.

Upstream — something before both wrecked them independently. The earlier block is the first casualty, not what took the later one.

Cascade — the earlier block's disruption consumed the later one's time.

Each gets a test. Every test is arithmetic in the database, never inference by the model.

Anchor control → tests upstream

Restrict to days the user's most reliable early block completed — the mornings that weren't lost. If the pattern weakens, something upstream was carrying it. If it holds or strengthens, it wasn't.

On this dataset it strengthened. Lift went from −53 to −55, and the afternoon still failed 0 of 11 days when the morning held against 6 of 11 when it didn't. Upstream ruled out.

Domain spread → tests bad-day effects

Compare the trigger against every later block, split by category. A bad day drags everything. A specific relationship doesn't.

Same-category lift −53. Everything else, −10. Breakfast and dinner did not move at all. The effect is specific to one kind of work.

Gap sensitivity → separates cascade from carry

If the effect weakens as the time between blocks grows, disruption is consuming the later block mechanically. If it holds across a gap, something is being carried rather than consumed.

Not enough pairs across enough gap ranges to say. Reported as untested — never as ruled out.

The engine eliminates. It does not diagnose. What survives is reported as what's left, never as what's true — and a test with insufficient data means untested, which is not the same as cleared.

LAYER 03 · NARRATE — REAL OUTPUT

What it produced

Unedited, from one month of real use. Every figure traced back to the database and verified.

STRUCTURAL

When Deep work morning holds, Deep work afternoon holds with it every time. That's not a lost start to the day — the same split is there on days the early routine landed. Breakfast and Dinner barely moved with it, so this stays with this kind of work.

Afternoon failed 0 of 11 days morning completed, and 8 of 15 days it didn't. On days the early routine landed, afternoon still failed 0 of 11 when morning held and 6 of 11 when it didn't.

PATTERN · DAY

Sunday is the one day the schedule consistently comes apart — fail rate of 51% against a best of 19% on Tuesday and Saturday. Every other day sits between 19% and 31%.

Sunday: 20 failed of 39 relevant instances. Tuesday and Saturday: 6 of 31 each.

STRENGTH

The early routine block is the steadiest one in the schedule — completed 26 of 29 tracked days. The reflections show it surviving disrupted nights and interrupted weeks.

26 completed against 3 missed across 29 days. Only 1 miss in the last 7 against 6 completions.

Every insight set includes at least one genuine, evidence-backed strength. Not consolation — a fact that happens to be good news.

THE HARDEST PART

Accuracy is kinder than encouragement.

Real reflection data is full of self-blame. Verbatim, from one month:

> *"Sloth."*

> *"Bad day."*

> *"Saturday, cut me some slack."*

That last one is a person pre-emptively arguing with an app they built themselves. This audience arrives already fluent in self-criticism. They do not need a mirror.

THE ENGINE MUST

Name structural findings — a mechanism, a slot, a missing boundary. State counts with their denominators. Include a real strength. Offer at most one small structural change. Be truthful about a bad week.

THE ENGINE MUST NEVER

Use the words lazy, failure, discipline, willpower, or "should have." Echo the user's own harshest phrasing back at them. Reassure about a judgment it isn't making — denying a verdict implies one was available.

The test every line must pass: would this still be all right to read during a week where almost nothing got done?

LAYER 04 · SURFACE

One useful thing, at the right moment.

Findings sit in the background and appear only where they change something. Not a dashboard. Not a chatbot you visit.

MORNING

A single card, once

The top finding, dismissed permanently once read. It returns when there is genuinely something new — roughly weekly, which is how often the app actually learns something.

THE MOMENT SOMETHING BREAKS

Recovery, not a verdict

Mark a block missed and the sheet names the mechanism instead of counting failures — and shows the user their own last written note on that block. If a real streak just broke, the diagnosis is suppressed entirely. That is not the moment for analysis.

BEFORE IT BREAKS

A cutoff that says why

A quiet notification partway through a block: what it ends at, what comes next, and what running over has cost before. It names the consequence and leaves the decision alone.

WEEKLY

The shape of the week

Seven squares, one number that is honest on a bad week, and the week's leading finding. The headline is days accounted for — never a completion percentage.

THE MECHANIC

The streak measures honesty, not perfection.

A completion streak breaks on the first bad day — which is the uninstall moment. So the protected number is different: a day counts when its blocks have been accounted for, whatever the outcome.

Marking something missed keeps the streak alive. Only silence breaks it.

THE SAME COMPLETION RATE, TWO DIFFERENT BEHAVIORS

Morning prayer block 14 missed · 1 unaccounted	Reported almost every time, including the failures. That is accountability.
Weights block 6 missed · 15 unaccounted	Silently abandoned. That is the drift itself.

The squares carry both facts at once. Filled height shows what was completed; the neutral band above it shows what was missed but reported. A fully accounted day fills the square regardless of outcome.

The missed portion is deliberately not red. A week of red squares is a wall of errors, and that is the signal this product exists to avoid.

Teal completed · neutral accounted for · empty unanswered. The sixth day was never closed out — that is the one that breaks the streak.

WHERE IT STANDS

Built today

The engine runs end to end. It captures, computes, narrates twice a week, and surfaces findings in three places.

- **Capture layer** — outcomes, quality, reasons, reschedule history, sleep and wake boundaries, nudge telemetry
- **Evidence engine** — one SQL function, integrity filters, block coupling with both signs, discriminating tests, caveats travelling inside the payload
- **Weekly narrator** — one AI call per user twice a week (Mon/Fri), up to five ranked insights including `structural`, stored atomically, never regenerated on read
- **Morning card, recovery injection, weekly review**
- **Cutoff nudges** — fires mid-block, cites the downstream cost, answerable from the lock screen

- **Accountability streak** and the two-tone weekly shape
- **Quality drift detection** — degradation caught before a block starts being missed

Not yet built

- **Intervention effectiveness** — reading back which nudges actually changed outcomes for this specific person
- **Intention reliability** — whether the user's plans are becoming more accurate over time
- **Drift detection** — noticing attention has left, without ever seeing where it went

WHAT IT BECOMES

An app that knows what works on you.

Today the engine explains what happened. Completed, it closes the loop — every nudge becomes an experiment, every response an outcome, and the system learns which interventions move this specific person.

> MONTH SIX
>
> "The cutoff at 10:30 has worked eleven of the last fourteen nights. The one at 9:00 never did — you dismissed it every time. Your mornings are now landing at twice the rate they were in July, and it started the week you stopped scheduling deep work after dinner."

Nothing in that paragraph is generated from a template. It requires six months of one person's interventions, responses, and outcomes — which is precisely why it can't be copied.

DRIFT DETECTION, PRIVATELY

The user selects which apps count as distraction. The selection is returned as opaque tokens — FlexMax is structurally incapable of knowing which apps were chosen. It learns only that attention left, for roughly how long, and what happened next.

INTERDEPENDENCE, NOT ISOLATION

Blocks are not independent. Naming the pair — "your workouts disappear in the same weeks your deep work slips; cutting one doesn't protect the other" — is a different and more useful finding than either block's own numbers.

WHY IT HOLDS

The model is replaceable. The record isn't.

1
AI call per user per week

100%
of arithmetic computed in the database

0
figures the model is permitted to invent

A competitor can ship a weekly "why your plans fail" card next quarter. What they cannot ship is six months of a specific person's interventions, responses, and outcomes — that requires that person's six months.

The switching cost compounds with every week of use. On day one the app knows nothing. On day ninety it knows that this person's evenings overrun, that they protect deep work above everything, and that their mornings are downstream of their nights.

STATED SO IT CAN BE FAILED

FlexMax must make execution measurably better, not merely make the user feel understood. If people love the findings but don't follow through any better, this is a sophisticated journal and the thesis is wrong.

Your schedule keeps failing. It doesn't know you.

FlexMax learns from what actually happens — including the parts you'd rather not report to a person.

The findings in this document are unedited output from one month of real use, verified against source data. Single-user validation — testing whether it generalizes is the next step, not a settled question.

Download as PDF

Interactive demo

Privacy policy