THE WORKSHOP

Event Storming a Process

The default Event Storming session and the one you'll run most often. One process, one wall, everyone who touches it in the room, three hours. You leave with a precise shared model of the flow and a short list of the questions it raised. Where Big Picture looks for shape across a whole domain, Process Level looks for precision within one flow.

2026-04-13
barkingiguana.com/writing/the-workshop-event-storming-a-process/

A facilitator's playbook -- print, annotate, and keep on the table.

Contents

Where Process Level sits 3

The Process Modelling palette 3

Intent 4

When to use it 4

Participants 4

Materials and timing 5

Facilitator playbook 5

Worked example — Pagebound's order-to-delivery flow 8

What can go wrong 9

Outputs 9

Where to go next 10

This is the second of three posts on running Event Storming. The Event Storming a Domain post is the entry point in Brandolini's ordering and introduces the technique at a whole-domain scale; if you haven't read it, start there. This post picks up where Big Picture drops off: a dot-voted hotspot from a Big Picture session is the natural scope of a Process Level session.

Event Storming an Architecture is the next step — zooming further in, turning a Process Level map into a software design. Coming soon.

For the technique in action inside a small startup, see Event Storming: Building Shared Understanding.

Where Process Level sits

Brandolini's three Event Storming levels, in order:

- **Big Picture** (the previous post) — the widest zoom. Whole domain, many people, days of work. Uses only three note colours (orange events, pink hotspots, yellow systems/people) because the goal is shape, not precision.
- **Process Level** — this post. One flow, small team, three hours. Adds commands, actors, policies, and read models to the wall; this is where Brandolini's full Process Modelling palette comes in.
- **Software Design** — one flow's code boundaries. Developers in the room. Three hours. Turns the wall into aggregates and bounded contexts, makes the policies binding, and names the places that need an anti-corruption layer.

Most of the time you run Process Level on its own, on a scoped flow — a billing cycle, a deployment pipeline, an incident that crossed a couple of services — without ever running Big Picture first. When you *do* run it after Big Picture, the scope comes from a dot-voted hotspot the Big Picture wall surfaced.

The Process Modelling palette

Brandolini's Process Modelling uses six note colours. Four of them carry the backbone of the wall; the other two appear where they add precision.

- **Orange** — events. Things that happened, past tense. The backbone of the wall. *"Payment Captured." "Stock Reserved."*
- **Blue** — commands. The intent that produced the event, present tense, imperative. *"Capture Payment." "Reserve Stock."*
- **Small yellow square** — actors. The person or system that issued the command. *"Customer." "Warehouse picker." "Stripe."*
- **Pink** — hotspots. Disagreements, questions, painpoints, anything the room flags for follow-up.
- **Lilac / purple** — policies. The *"whenever X, then Y"* rules that tie an event to the next command. Most of a Process Level wall's policies are implicit (events quietly triggering commands) — but when the room agrees on a rule out loud, or when a rule is contested and then resolved, it earns a purple sticky.
- **Pale green** — read models. The data a policy (or a person) consults before deciding what to do. *"Before reserving stock, check current stock level."* Green notes live next to the policy or command that reads them.

If you're running your first few Process Level sessions, it's fine to stay on the four-colour backbone (orange, blue, yellow, pink) and capture policies verbally in the notes. Brandolini's full palette is what you grow into as the room gets comfortable — not a gate you have to pass before you run the session.

Intent

Build one precise shared model of one specific process — a flow, a pipeline, a cycle, an incident, a feature area — with the people who touch it in the same room, so the team building or operating that process has one model, not five.

The output is a wall of events in time order, commands under them, actors above them, and a prioritised list of the questions the session raised.

When to use it

Reach for Process Level when:

- You're about to build a specific flow and the team's mental models of it are quietly different
- You're inheriting a process nobody documented
- You're investigating an incident that crossed two or three services
- You've zoomed in from a Big Picture session and you have a named hotspot to dig into
- A piece of work is about to cross multiple people's areas and you want to spot mismatches early

Don't reach for Process Level when:

- The scope is the whole business or a whole product line — run Big Picture first
- You're ready to design code — run Event Storming an Architecture
- Only one team is involved and they share a strong mental model already
- The scope is one screen, one function, or one isolated job — it's too small for the ceremony

Participants

Facilitator. Does not participate in content; their job is to manage the session. Ideally someone who's run one of these before; if not, pair with someone who has.

Domain expert(s). The people who know how the process actually works. For a billing flow, the finance lead. For a deployment pipeline, the SRE who runs it. For an incident review, the engineers who responded. One or two, not a crowd.

Developers. At least one, and include a junior if you have one. Juniors ask the questions seniors have stopped asking.

Product or design — whoever will turn the output into stories.

Operations, support, frontline. For incident, deployment, or support-heavy flows, *these are the domain experts*. Don't tuck them in as afterthoughts.

Group size: 4–8. Smaller than Big Picture because the scope is tighter. Fewer than four and the conversation is too thin; more than eight and the voices overlap.

Who to leave out:

- **End users and customers.** People self-censor around the people they serve. Interview users separately; bring their words in on a sticky note.
- **Senior leaders who can't stop correcting.** If the senior *is* the domain expert, brief them first: their job is to answer when asked, not to lead.
- **Spectators.** Anyone "just observing" absorbs airtime without contributing. Either in or out.

Materials and timing

Phase	Duration	Materials	Key question
Arrivals, intro, ground rules	~15 min	–	"What are we doing and why?"
Chaotic exploration	20 min	Orange	"What happens?"
Timeline	30 min	Orange + pink	"What order? What's wrong?"
Break	10 min	–	–
Commands and actors	20 min	Blue + yellow (purple + green as rules emerge)	"What triggered it? Who did it?"
Hotspots	30 min	Pink	"What scares us most?"
Wrap-up, owners, next steps	15 min	–	"Who owns what next?"
Buffer	20 min	–	–
Total	**~2h 40min inside a 3-hour block**		

The four working phases are 100 minutes. The remaining hour is the unglamorous stuff: arrivals, the intro, the mid-session break, the wrap-up, and the conversations that inevitably run long. Don't try to fill the 20 minutes of slack – you'll need it.

Facilitator playbook

Phase 1 – Chaotic exploration (20 min)

Before the first sticky goes up, do two things that look trivial and aren't.

Set the safety out loud:

> *"The only rule for this phase is that every note is valid. Duplicates are fine, half-formed ideas are fine, things that might be wrong are fine — that's exactly what we're here to find. If you're not sure, write it anyway."*

Set the granularity. Stick two example events on the wall yourself at the level a domain expert would say them out loud:

> *"Payment Submitted." "Parcel Dispatched." "Alert Fired." "Deployment Rolled Back."*

Hand out orange pads. Name the most junior person in the room:

> *"[Name], what's the first event you can think of for this flow? Doesn't have to be the start — just the first one that comes to mind."*

Then give the instruction that covers the rest of the phase:

> *"Write silently. Get everything you can think of onto orange notes and onto the wall. Don't worry about order. Don't worry about duplicates. Twenty minutes on the clock, then we stop."*

No talking until the timer goes off. By the end you should have 40–80 notes. Some will be duplicates; some will contradict; some will make no sense yet. That's exactly right.

What to watch for:

- **Someone talking instead of writing.** Gently: *"Get it on a sticky note."*
- **Someone waiting for permission.** *"Duplicates are gold. Write yours anyway."*
- **One person filling the wall while others have three notes.** Usually sorts itself out in the timeline phase; keep an eye on it.
- **Someone reaching for a pink note already.** *"Good instinct — hold that thought."*

Phase 2 — Timeline (30 min)

Now everyone talks. The job is to arrange the orange notes left-to-right in chronological order.

> *"Let's put these in order. Talk to each other. If you disagree, put a pink note on it and we'll come back."*

At the fifteen-minute mark, the wall will look messy and you'll worry. That is what success looks like midway through. There'll be clumps where people stood; gaps where nobody's arranged yet; two notes stacked because someone tried to merge them and gave up; overlapping candidates for the first event; a couple of pink notes nailed into contested spots. If the wall looks tidy at the fifteen-minute mark, either the scope was too small or one person is doing all the moving.

Merge obvious duplicates. Leave ambiguous ones — if two notes *might* be the same event, that's a conversation worth having later.

What to watch for:

- **One person placing every note while everyone else watches.** The most common failure mode. Pair a developer with the domain expert and ask them to walk a section together.
- **No pink notes appearing.** Disagreements are hidden, not absent. Prompt: *"Is anything on this wall surprising you?"*
- **Rabbit holes into solution design.** *"Great implementation idea — park it. We're mapping, not building."*

- **Parallel flows emerging.** Let them spread vertically into swim lanes. A rollout flow and a rollback flow can share a wall.
- **Events causing events.** Someone asks *"so does Payment Captured cause Stock Reserved?"* Name the rule: *"Events don't cause events. Something reads Payment Captured — a person, a rule, a clock — and decides to reserve stock. The chain is always event → decision → command → event, not event → event."* First-timers want to draw arrows between orange notes within the first hour. Name the rule before they do.

Phase 3 — Commands and actors (20 min)

Hand out blue and yellow notes. Introduce them one colour at a time — if you drop both on the table at once, people grab whichever is closest and the wall gets noisy.

Blue first — commands. For each event, what intent produced it?

> *"Blue notes go below the orange event. They're the command that made it happen. 'Submit Payment' caused 'Payment Submitted'. 'Pick Order' caused 'Items Picked'. Every event has a command somewhere — even if the command is a scheduled job or a reaction to another event."*

Yellow next — actors. Who issued the command?

> *"Small yellow squares go above the command. An actor is a person, a role, or a system. 'Customer' is fine; 'the system' is not — which system? 'Warehouse picker.' 'Stripe.' 'Nightly cron.' Be specific enough that the name points to someone or something you could actually talk to."*

Two discipline points most first-time facilitators miss:

1. **Small yellow squares, not full-width notes.** The actor sits next to or on top of the command; it's smaller than the command, because the command is the important bit at this level.
2. **Deduplicate.** If the same actor issues three commands in a row, you don't need three yellow squares — stick one next to the first command and let the row speak for itself. Real ES walls have one or two actor squares per band, not one per event.

What to watch for:

- **"The system" on too many yellow squares.** *"Which system? Automated or manual? What happens when it fails?"*
- **Business rules hiding inside a command.** *"Wait, this command only runs sometimes — what decides?"* If the room can name the rule out loud, purple-note it in *"whenever X, then Y"* form between the triggering event and the command; if the decision needs a fact (a balance, a stock level, a flag), stick a pale-green read model next to the policy. If the rule is contested, leave it as a pink hotspot for the next phase to resolve.
- **One person's name on multiple recurring actors.** Scaling bottleneck. Pink note.
- **Commands that nobody can explain.** *"Who decides this?"* followed by silence is extremely valuable. Pink note.

Phase 4 — Hotspots (30 min)

Gather the pink notes — the ones you've been accumulating on the wall, plus new ones you'll generate by prompting for them. These are the most valuable output of the session.

The mechanics matter more than most facilitators realise; clustering pinks under time pressure is where first-time facilitators freeze. A shape that works:

1. **Read the pinks aloud, one by one (5 min).** You walk the wall and read each pink note. No commentary — just read. This refreshes the room and gives you time to spot repetition.
2. **Move notes into rough piles (10 min).** Take the pinks off the wall and put them into 3–7 piles on a table or a clear section of wall. Let the room help. If a note could go in two piles, put it in the bigger one. The goal isn't clean boundaries; it's rough themes.
3. **Name each pile (5 min).** For each pile, write a one-sentence theme on a fresh pink note and put it on top. *"Rules nobody has written down." "Cross-team handoffs with no SLA." "Edge cases we deferred."*
4. **Owner and next step per pile (10 min).** *"Who owns finding the answer? What's the next step?"* Write both on the theme note. Time-box 90 seconds per pile; if a conversation runs long, park it.

Don't try to solve anything in-session. Identify, name, assign, move on. Solving is not this phase's job.

Worked example — Pagebound's order-to-delivery flow

Pagebound is a mid-sized online independent bookshop: about 200,000 customers, six warehouses, an engineering team of thirty, a customer support operation that fields returns and lost parcels. A recent Big Picture session on the whole customer experience produced a prioritised list of hotspots; the top one was *"when do we reserve stock?"* — commerce said at checkout, the warehouse said at payment captured, and both teams had been operating on their own model for eighteen months.

The Process Level session scope, in one phrase: **the order-to-delivery flow, from checkout started through parcel delivered.** Six people in the room — commerce lead, a commerce engineer, the fulfilment team lead, a warehouse supervisor, the SRE who owns the payment integration, and a customer-success lead who's been fielding the over-sold complaints.

Here's what the wall looks like at the end of the session — fifteen events in rough time order, commands underneath, small yellow actor squares deduplicated per band, one purple policy and its pale-green read model where the room agreed on a contested rule, and four pink hotspots showing the questions the session raised:

Notice what's on the wall and what isn't. The actor bands show that the customer initiates the first two events, then the order service quietly does its work, then the warehouse takes over for three events, then the carrier, then the customer again at the end. That's five handoffs — and every handoff is a candidate for something to go wrong, which is why three of the four pink notes cluster around handoff boundaries.

One purple policy made it onto the wall, and its presence is the whole point of the session: the room resolved the stock-reservation timing question out loud, decided that *"whenever Payment Captured → reserve stock"* was the right rule, and stuck it up so nobody could quietly forget later. A pale-green read model beside it names the fact the policy depends on — *current stock level* — because the next thing the team will argue about is what happens when that number is zero, and pinning the read model now saves

an argument later. The other implicit policies (events quietly triggering subsequent commands) are left unspoken — that's fine for Process Level; the Architecture session is where every crossing gets a policy and every policy gets its read model.

The four pink hotspots are the real output. Two (over-sold orders, substitution) will turn into Example Mapping sessions with business rules attached. One (the handoff gap) becomes an investigation with the carrier. One (wrong-address SLA) becomes a conversation between customer success and legal. None of them get "solved" at this session, and trying to would burn the next two hours on arguments that belong in their own meetings.

What can go wrong

Named failure modes.

The silent room. Nobody is writing or talking. *Recovery:* The prompt is too abstract. Make it concrete: *"What's the first thing that happens when a customer clicks 'place order'?"* *Stop if:* 20 minutes in and the wall is still empty. Scope is wrong, people are wrong, or there's a political problem you haven't named.

The lecture. One expert explains while everyone listens politely. *Recovery:* Pair people up, give each pair a section of the wall. *Stop if:* Two pairs in and it's still the same voice. The session is producing one person's model.

The argument. Two people disagree about how something works. *Recovery:* Let it play for 2–3 minutes. This is often the session working. If it's not resolving, pink note it. *Stop if:* The argument has gone personal. Break; resume only if the air has cleared.

The solution-jumper. Someone keeps designing the code instead of mapping the process. *Recovery: "Great implementation idea — park it."* *Stop if:* They can't hold the distinction after a third prompt. They belong in an Architecture session, not this one.

The missing person. Nobody in the room knows how a key part of the process works. *Recovery:* Pink note it with a name. *"Need to talk to [person] about [topic]."* *Stop if:* Multiple key parts are owned by people not in the room. Reschedule with the right attendees.

The political silence. A senior is in the room and the juniors have stopped writing. *Recovery:* Pair juniors with peers away from the senior; or ask the senior to step out for a call (briefed in advance); or enforce silent writing with no exceptions. *Stop if:* None of the above shifts the dynamic. Photograph what's on the wall, reschedule.

Outputs

Same day, 24 hours:

- Panoramic high-resolution photographs of the wall.
- A transcribed event list, command list, and hotspot list (each pile named, owner, next step) in a shared document.
- A short summary to participants: *"Here's what we found, here's what happens next."*

The product owner's (or equivalent's) week:

- **Turn each event into a vocabulary entry.** *"Stock Reserved"* means exactly one thing; defend the phrase against drift.
- **Triage the hotspots.** Each pile becomes one of: (a) work for this sprint, (b) a time-boxed investigation, (c) a follow-up workshop (Example Mapping, Decision Tables, Architecture), (d) a conversation. Resolve the ones that block the next sprint; defer the rest.
- **Book the follow-ups.** Don't let momentum dissipate.
- **Walk the wall with anyone who couldn't attend.** Their perspective often surfaces hotspots the original room missed.

Where to go next

- **Event Storming a Domain** — zoom out when you realise the scope of your Process Level problem is actually organisational, not procedural.
- **Event Storming an Architecture** — the natural next step when the Process Level flow is clear and you're about to turn it into code boundaries.
- **Event Storming: Building Shared Understanding** — the narrative version on a smaller team.

About this playbook

This playbook is part of *The Workshop*, a reference series of facilitator playbooks published at barkingiguana.com. The canonical, up-to-date version lives at barkingiguana.com/writing/the-workshop-event-storming-a-process/.

These posts are LLM-aided. Backbone, original writing, and structure by Craig. Research and editing by Craig + LLM. Proof-reading by Craig.