THE WORKSHOP

DDD Modelling

Turn a clarified Event Storming Architecture wall into a context map: named bounded contexts, aggregates with stated invariants, value objects, and integration patterns at every crossing.

DURATION

3h 45m

GROUP SIZE

—

people

WHAT YOU BRING

A clarified Architecture wall (or agreed flow with crossings marked), a list of external systems, and the vocabulary the team argues about.

WHAT YOU LEAVE WITH

- Aggregate cards, each justified by one stated invariant
- Named bounded contexts with committed vocabulary
- Integration pattern picked and labelled at every crossing
- One-page context map plus owned, dated ACL spikes

WHO TO INVITE

- **Facilitator (DDD-fluent).** Keeps vocabulary straight and sorts aggregate vs entity confusion; pair with a tech lead if generalist.
- **Developers and architects.** The four to six people who will write the code; aggregates and boundaries must land in their hands.
- **Domain expert with veto.** Stops elegant models that don't match how the business actually works; postpone the session without them.
- **Product lead.** Turns outputs into backlog shape and carries the context names into future product conversations.
- **Adjacent-team tech lead (optional).** Prevents the context map being drawn unilaterally and blocks integration patterns their team can't support.

USE WHEN

An Architecture event-storm has happened and code is about to be committed against boundaries

A monolith is being split and the seams need names, not gestures

Two contexts have started leaking vocabulary and an integration pattern must be picked

The ubiquitous language has drifted and a word means different things to different people

AVOID WHEN

No agreed flow yet — run Event Storming an Architecture or Process Level first

The domain is unfamiliar to half the room — DDD becomes a hammer in search of nails

You're at container/component level — that's C4 Modelling territory

No domain expert with veto power is available to attend

THE WORKSHOP · DDD MODELLING

How the session runs

Phase 1 — Orient to the wall (15 min)

Walk the Architecture wall aloud, pointing at every aggregate candidate, boundary and crossing, and invite corrections. Accept small fixes; if the wall has genuinely drifted, stop and rerun Architecture first.

Phase 2 — Name aggregates and invariants (45 min)

For each candidate, write a card with a name and the one invariant that justifies it. Reject wishful rules, split god aggregates, and aim for five to eight cards in a single-context session.

Phase 3 — Identify value objects (20 min)

Walk each aggregate and ask whether each concept has persistent identity or is defined purely by its values. Park shared value objects (Money, Address) in a neutral area as shared-kernel candidates.

Phase 4 — Draw bounded contexts (30 min)

Draw a thick line around each cluster of aggregates that shares a vocabulary and name the context on a card inside. Test boundaries by asking whether key words mean the same thing on both sides.

Phase 5 — Mark crossings and pick patterns (45 min)

For every boundary-crossing line, pick an integration pattern from the named menu and write it on the crossing. Name the direction of dependency, the shape of the crossing, and the pattern's cost out loud.

Phase 6 — Anti-corruption layer decisions (20 min)

For each ACL marked, write their-words versus our-words columns and fill in the translation. Name an owner for every ACL so unowned translations don't quietly drift into the model.

Phase 7 — Write the context map (15 min)

Draw every context as a labelled box and every crossing as a labelled arrow on a single page. One page is the artefact; mark any aspirational patterns as planned rather than over-claiming.

Phase 8 — Commit next spikes (15 min)

Close on three or four concrete commitments — first ACL built, package renames, follow-up sessions, adjacent-team review — each with a named owner and a date. Commitments, not summaries, end the session.