INTERACTION DESIGN CASE STUDY

Report a Claim

Redesigning State Farm's Online Claims Intake, Phase 2

Detailed interaction design for a multi-line-of-business claims flow — mapping every deviation point a user could hit between "start" and "submit," and resolving the edge cases that make or break a task-critical form.

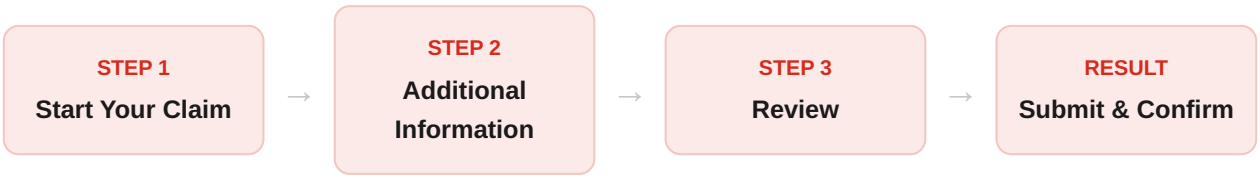
AGENCY	CLIENT	ROLE	DELIVERED
Razorfish	**State Farm**	**Interaction Design**	**December 2012**

01 — THE CHALLENGE

One form, four lines of business, dozens of exit ramps

State Farm's "Report a Claim" flow needed to work across four very different lines of business — Auto, Health, Life, and Property — while still feeling like a single, coherent three-step process. That constraint drives most of the hard design problems here: a form generic enough to flex across wildly different claim types, but specific enough to catch duplicate reports, protect data integrity once a claim is underway, and gracefully handle a user who can't finish in one sitting.

This phase of the work picked up an existing conceptual model and pushed it into full interaction detail — every possible action a user could take at every step, where it could occur, and what it led to — so that business stakeholders and the build team were working from the same shared map of the system before a single screen was finalized.



> **Design principle at the core of this work:** separate what the system does from what the interface shows. The brief explicitly distinguished conceptual diagrams (the domain and overall interaction), UX flows (how a user actually moves through it), and wireframes (the screen-level detail) — three layers of precision built in sequence so nothing got specified twice, inconsistently.

02 — MAPPING THE POSSIBILITY SPACE

Nine actions, every one of them a decision point

Before any screen was drawn, the team catalogued every possible action a user could take anywhere in the flow, and the exact step(s) where each one was available. That single reference model — what's possible, and where — became the shared source of truth for the rest of the design.

ACTION	WHAT IT DOES
Help Me Choose	Assists the user in picking the right line of business — a distinct, lighter-weight pattern from the general DC-6 help module
Dupe Check	Flags a possible duplicate claim already on file and lets the user drill into details
Point of No Return	Warns the user that Step 1 information becomes locked once they continue
DC-6 Help Module & FAQs	Surfaces the 800 number and a link into the FAQ library from anywhere in the flow
Cancel	Exits claim creation or editing, with a confirmation step to prevent accidental loss
Save & Continue Later	Persists in-progress work for 14 days so the user can resume elsewhere
Skip Remaining Questions	Jumps straight to Review, trading completeness for speed
Edit	From Review, lets the user jump back and revise a prior answer
Submit	Finalizes the claim and routes to a confirmation state

03 — SIGNATURE PATTERN: HELP ME CHOOSE

Three ways to answer the same question

The very first decision a claimant makes — Auto, Health, Life, or Property — is also the one with the highest cost of getting wrong, since it determines everything downstream. Rather than betting on a single help pattern, the design specified three destinations for the same "Help me choose" trigger, so the team could test which fit best without re-architecting the flow:

OPTION A

In-page contextual help

Expands inline, directly beneath the question, with a short description for each line of business — no navigation away from the form.

OPTION B

Linked FAQ entry

Deep-links into the DC-6 FAQ page with the relevant question already expanded, reusing existing help content rather than duplicating it.

OPTION C

Dedicated info page

Sends the user to a standalone page with a clear "Back" path that returns them to their exact place in the form.

04 — SIGNATURE PATTERN: DUPLICATE DETECTION

The hardest problem in the flow

Duplicate claims are costly and confusing on both sides — for State Farm's processing systems and for a customer who doesn't realize they, or someone else, already reported the same incident. This pattern got the most design iteration of anything in the document, because it had to handle several very different situations gracefully:

REGISTERED PARTICIPANT, ONE LIKELY DUPLICATE

Inline notice → detail comparison → resolve

An inline message appears the moment the system detects a likely match. The user can open a side-by-side comparison — highlighting specifically what's different between "today's claim" and the original (e.g. date/time of incident) — and choose to switch to the existing claim or proceed with a new one.

NON-PARTICIPANT, CONFIRMED DUPLICATE

Hard stop, not a soft warning

A user who isn't a registered participant on the existing claim can't be shown its details for security reasons. Instead the flow terminates outright, with a clear explanation and a phone number — deliberately not offering the option to save progress, since there's nothing legitimate left to save.

RETURNING USER, UNSUBMITTED CLAIM FOUND

A softer, non-blocking variant

When the "duplicate" is really the user's own previously-started, unsubmitted report, the tone shifts from warning to reminder — "double check your information before proceeding" rather than a hard interruption — with a simple Switch/Ignore choice.

05 — DESIGNING FOR THE UNHAPPY PATHS

Save & Continue Later, and everything that can go wrong with it

Letting a busy claimant save partial work and come back later sounds simple until you design every state it can land in. This pattern alone required four distinct screens beyond the basic save confirmation:

Happy path

- User names the saved report (kept under 20 characters)
- Confirmation screen states the 14-day window explicitly
- A follow-up email repeats retrieval instructions in case the user closes the tab

Failure & limit states

- **Save limit reached:** a user with 5 existing saved reports is warned before starting a 6th, and prompted to delete or submit an old one first
- **Technical failure:** if saving fails outright, the user is offered a clear choice — keep going now, or exit — rather than a dead end

Point of No Return, handled honestly: Once a user continues past Step 1, that information can't be self-edited again — only an agent can change it. Rather than burying that constraint in fine print, the design surfaces it as an explicit confirmation step: "Is everything correct?" with a plain-language explanation of why, before the user can lock it in.

06 — WORKING THROUGH OPEN QUESTIONS

What the design surfaced back to the business

Good interaction design specs don't just answer questions — they surface the ones the business still needs to make a call on. Several genuine open questions were carried through the document at the exact decision points they affected, so stakeholders could resolve them with full context rather than in the abstract.

Scope of duplicate checking: should the system check only against existing claims, or also against informal loss reports? Resolved to claims only.

Multiple duplicates found: if more than one possible match exists, should the user — participant or not — be routed straight to the access-transfer dead end? Confirmed yes.

Canceling a retrieved report: should canceling mid-edit revert to the last saved version, or discard entirely? Flagged for further discussion rather than assumed.

Disabled vs. hidden controls: for bottom-of-page buttons that can't be used until required fields are complete, should they be hidden outright or visible-but-disabled? Left open for the visual design and dev teams to weigh in.

07 — WHAT THIS CASE SHOWS

Takeaways for portfolio review

1. **Catalog the possibility space before designing screens**

   Mapping every action and its valid locations up front (Section 02) meant later screens were built against a complete, agreed-upon model — not discovered ad hoc as edge cases emerged in review.

2. **Match the interruption to the actual stakes**

   Three distinct duplicate-detection treatments — soft reminder, detailed comparison, hard stop — show real restraint: not every "duplicate" deserves the same level of friction.

3. **Design the failure states, not just the happy path**

   Save limits, technical save failures, and locked-field warnings were treated as first-class screens, not afterthoughts bolted on during QA.

4. **Leave genuine open questions visible, not hidden**

   Flagging unresolved business decisions directly in the deliverable — rather than quietly picking an answer — kept stakeholders and the design honest about what was actually settled.

This case study is shared with permission and summarizes and reframes original interaction design work (Razorfish for State Farm, "Report a Claim — Phase 2 Interaction Design Artifacts," December 2012) for portfolio presentation. Specific proprietary UI copy and internal reference materials have been paraphrased or omitted.