ArcGIS Case Studies

ON THIS PAGE

ArcGIS Case Studies

Three fictional-but-realistic organizations worked end to end: a land trust, a coffee shop site selection, and a rural utility's storm response. Every decision cites the chapter that governs it.

Case Study: A Land Trust Maps Its Preserves

The Millbrook Valley Land Trust is a three-person conservation nonprofit holding fourteen preserves spread across two counties. Over one working season, its staff went from a filing cabinet of deeds and photocopied trail maps to a complete digital inventory: authoritative preserve boundaries assembled from county parcel data, a digitized trail network, a volunteer-run observation layer in Field Maps, a public StoryMap, and a board-ready print atlas produced in ArcGIS Pro. This case study narrates the decisions in the order they were made, including the one that went wrong. The organization and every person named here are fictional composites assembled

from common small-nonprofit patterns; all numbers are plainly illustrative, not measured data from a real project.

The starting point

Three staff. Marisol is the executive director and answers to the board. Dana is the stewardship coordinator and became the accidental GIS person, the way stewardship coordinators usually do. Ruth runs volunteers and community outreach. Their spatial data, such as it was: deed descriptions in manila folders, a shapefile a consultant delivered years earlier with no metadata and no memory of how it was made, hand-drawn trail maps of varying honesty, and GPS tracks from Dana's phone.

The first decision was where the system would live. Dana chose an ArcGIS Online organizational subscription through Esri's nonprofit program, with ArcGIS Pro installed on one workstation, over a desktop-only setup with files on a shared drive. The reasoning was about audiences, not software: nearly everyone who needed to see this data was not staff — volunteers standing in a preserve, board members at a meeting, the public deciding whether to visit. A web-first platform reaches those people without exporting and emailing files. The product landscape, account types, licensing tiers, and what actually consumes credits are the subject of Compendium Chapter 2 (The ArcGIS Ecosystem); Dana read the credits material first, learned that hosting modest vector layers is cheap while some analysis and geocoding operations are not, and stopped worrying about routine work.

The second decision, made the same week, was quieter but shaped everything after it: every dataset would have exactly one authoritative copy, and every map, app, and story would reference that copy rather than hold a duplicate. If you take one habit from this case study, take that one.

Assembling boundaries from county parcels

Dana's first instinct was to trust the consultant's shapefile. The decision, after a day of inspection, was to rebuild the boundaries from county parcel data instead. The reasoning: the county parcel fabric is continuously maintained and legally derived, while the shapefile had no documented source, no creation date, and — when overlaid on current parcels — visible slivers of disagreement along two preserve edges. The

alternative of digitizing metes-and-bounds descriptions straight from the deeds was rejected as a surveyor's job wearing a GIS costume.

Both counties publish parcels through open data portals. Evaluating a source before you build on it — who maintains it, how often it updates, what the attributes actually mean — is Compendium Chapter 5 (Finding Data). Dana downloaded each county's parcels, then selected the parcels matching the trust's name variations and the parcel numbers recorded in the deed files. That surfaced two discrepancies worth the exercise on their own: one preserve was still recorded under a predecessor organization's name, and another parcel had been administratively split by the county since acquisition. Both were resolved against the deeds before any geometry was touched.

Projection came next, because each county delivered data in its own State Plane zone in feet. Dana projected everything into a single coordinate system for the working geodatabase in Pro and let the platform handle the web map projection when layers were published. Knowing why that step exists — and why "the layers don't line up" is almost always a coordinate system story — is Compendium Chapter 3 (Coordinate Systems and Projections).

With clean selections in hand, Dana dissolved each preserve's parcels into a single boundary polygon using geoprocessing tools in Pro — running tools deliberately, with environments checked, is Compendium Chapter 22 (Geoprocessing in Depth), and the broader menu of ways to create data is Compendium Chapter 11 (Creating Data). The decision that mattered inside this step: keep the source parcel identifiers in a field on the dissolved boundaries, and archive the raw parcel selections as their own layer. Boundaries derived from parcels are only as trustworthy as your ability to trace them back. A year later, when a county adjusted a parcel line, that field answered "which preserve does this affect?" in seconds.

Designing the schema before publishing anything

Before publishing a single layer, Dana wrote the schema on paper: preserve name, acquisition year, acreage computed from geometry rather than typed by hand, public access status as a coded value domain, a stewardship notes field, and a field for those source parcel IDs. Domains got particular attention — an access field constrained to a short pick-list means editors choose "Open — dawn to dusk" from a menu instead of

inventing eleven spellings of it. Field types, domains, and the discipline of designing before you collect are Compendium Chapter 12 (Schema Design). Dana also enabled editor tracking on every layer at creation, on the theory that knowing who changed what costs nothing now and is unobtainable later.

The layers were published as hosted feature layers; publishing paths, item settings, layer views, and performance behavior are Compendium Chapter 10 (Hosted Feature Layers). One more habit began here: every Friday, Dana opened each layer's item page and exported it to a file geodatabase, downloading the export to the office machine. It looked like busywork. Hold that thought.

Digitizing the trails

The trails came from two imperfect sources: Dana's phone GPS tracks, which wander under tree canopy, and high-resolution imagery, which shows the actual trail tread only where the canopy opens. The decision was to digitize in ArcGIS Pro against an imagery basemap, treating the GPS tracks as a guide rather than gospel — following the visible tread in open ground and letting the track carry the line through dense cover. Editing templates, snapping, and the mechanics of clean digitizing in both Pro and the browser are Compendium Chapter 13 (Editing Workflows in Web and Pro).

Snapping was non-negotiable: trail segments snap to junctions, junctions sit on the trail they branch from, and loops actually close. Attributes followed the same schema discipline as the boundaries — trail name, a difficulty domain, surface type, blaze color. Fourteen preserves took several sessions spread across a few weeks, which is worth saying plainly: digitizing is honest, tedious work, and pretending otherwise leads to schedules nobody can keep.

The volunteer observation layer

Ruth's ask was simple: volunteers walk these preserves constantly — let them report what they see. The decision was a dedicated point layer with a deliberately tight schema — an observation type domain (invasive plant, downed tree, trail damage, boundary encroachment, wildlife sighting), a notes field, photo attachments — delivered through ArcGIS Field Maps on volunteers' own phones. Dana configured the form in Field Maps Designer so required fields stayed minimal; a volunteer standing in the rain will not type paragraphs, and a form that takes under a minute gets used.

Because several preserves have no cell coverage, Dana defined offline map areas so the app works disconnected and syncs later. Form design, offline workflows, and the rest of the field-app family are Compendium Chapter 30 (Field Maps, Survey123, QuickCapture).

For accounts, Ruth resisted the tempting shortcut of a shared login and instead added each volunteer with a user type suited to field editing. Shared logins destroy editor tracking and make offboarding impossible; user types, groups, and member management are Compendium Chapter 34 (Administration).

The misstep: one layer, too many hands

Here is the mistake, and it was Dana's. To get the pilot moving fast, Dana shared the master observations layer — the editable, authoritative one — directly into the volunteer group with full editing enabled, and added the same layer to the draft public web map so Ruth could preview how observations would look on the trust's website.

Two problems arrived in the same week. First, a conscientious volunteer, tidying what he believed were his own duplicate points, deleted several other volunteers' observations — full editing meant every volunteer could edit and delete everyone's features. Second, Marisol opened the draft public map and found every unreviewed report on display, including a volunteer's sighting of a rare plant, precisely the kind of location a land trust must not broadcast.

The recovery had two halves. Editor tracking — enabled back in the schema-design week — turned a mystery into a ledger: it showed exactly which features were deleted, by whom, and when, so nobody had to interrogate volunteers on vibes. And the Friday export habit paid for every minute it had ever cost: Dana pulled the previous week's file geodatabase copy, extracted the deleted points, and appended them back after checking for anything newer. Total loss: a few days of edits at most, and one apologetic-but-relieved phone call to a volunteer who had done nothing worse than trust the interface he was given.

The structural fix was hosted feature layer views, which is what Dana should have built on day one. The master layer went private, staff-only. From it, Dana created two views from the layer's item page: a volunteer view with editing constrained so contributors add features and modify only their own, and a public view that is read-only, filtered to

staff-reviewed observations, stripped of sensitive-species records, and exposing only the fields the public needs. Views are cheap, they inherit the master's data live, and they let one dataset serve three audiences with three different levels of trust — the full treatment is in Compendium Chapter 10 (Hosted Feature Layers), and the symptom-to-cause reasoning that untangled the week is the working style of Compendium Chapter 39 (Troubleshooting Encyclopedia). The lesson generalizes far beyond land trusts: never hand any audience the master layer. Hand them a view shaped like their job.

The public StoryMap

With the layer architecture sane, Ruth built the public face: a StoryMap touring all fourteen preserves — a photograph, a short history, access details, and an embedded map for each. The load-bearing decision: every embedded map is a web map built on the read-only public views, not an express map drawn by hand inside the story. The reasoning is the reference-don't-copy rule again — when a boundary shifts or a trail reroutes, the StoryMap corrects itself without anyone remembering it exists. StoryMap structure, narrative pacing, and embedding choices are Compendium Chapter 27 (StoryMaps). The maps themselves got deliberate styling — muted boundary fills, trails symbolized by difficulty — using the approaches in Compendium Chapter 7 (Styling and Smart Mapping), and pop-ups trimmed to name, access status, and a photo, per Compendium Chapter 9 (Pop-ups, Fields, and Labels).

The board atlas from Pro

Marisol's request closed the season: a printed map book for the board meeting, one page per preserve. The decision was a spatial map series in a single ArcGIS Pro layout, using the preserves layer as the index, rather than fourteen hand-assembled layouts. Build one good page — locator inset, scale bar, north arrow, acquisition details pulled from the layer's own attributes — and the series generates the other thirteen, each centered and scaled to its preserve. Layouts and map series are Compendium Chapter 24 (Layouts and Map Series); organizing the project and maps that feed the layout is Compendium Chapter 21 (Pro Interface and Projects); and the print symbology — heavier labels, a restrained basemap, colors that survive a grayscale photocopier — is Compendium Chapter 23 (Symbology and Labeling). One PDF export, fourteen pages, an afternoon of polish after the first page looked right. When boundaries change next year, the atlas is a re-export, not a rebuild.

The decisions that mattered

- **Rebuilding boundaries from county parcels, with parcel IDs retained on every polygon**, instead of trusting an undocumented shapefile — lineage turned every future boundary question into a lookup instead of an investigation.
- **Designing the schema — domains, computed acreage, editor tracking — before publishing**, when changes cost nothing, rather than after apps and volunteers depended on the layers.
- **One authoritative layer, many views**: volunteers, the public, and staff each got a view shaped to their role and trust level — a rule adopted the hard way, one deleted-points incident too late.
- **The weekly export habit** — unglamorous, faithfully kept — which converted a deletion incident into a few days of lost edits instead of a permanent hole in the record.
- **Reference, never copy**: the StoryMap, the web maps, and the print atlas all point at the same authoritative layers, so a boundary fixed once is fixed everywhere, forever, for free.

Case Study: Site Selection for a Third Coffee Shop

Marisol Vega owns Cardinal Coffee, a two-location cafe business in the mid-sized city of Rivermont. Both shops are profitable, her leasing broker has sent her six candidate storefronts for a third location, and she has about three weeks to narrow the list to two finalists worth in-person visits and lease negotiations. She has no GIS staff and no analysis budget beyond an entry-level ArcGIS Online subscription, so everything in this case runs in a browser: Map Viewer, Living Atlas layers, and a small handful of credit-consuming tools used deliberately. Cardinal Coffee, Rivermont, Marisol, and every number in this piece are fictional composites assembled to illustrate a realistic small-business workflow; no real organization, person, or dataset is depicted, and all figures are plainly illustrative.

Framing the question before opening the software

Marisol's first decision happened before she logged in. She wrote one sentence on a sticky note: "Which two of the six candidates have the strongest nearby customer base that my existing shops do not already serve?" Her reasoning was that a vague goal like "find the best location" invites endless map-making, while a sharp question tells you exactly which layers you need and, just as important, which ones you can skip. The question implied four ingredients: her own shops, the six candidates, her competitors, and the people who live near each site. It also implied a constraint that would matter later — "do not already serve" means the analysis has to account for overlap with her existing trade areas, not just raw market size.

She also decided up front to treat credits like cash, because on a small subscription they effectively are. Geocoding, drive-time generation, and data enrichment all consume credits, and the ArcGIS credit model — what burns credits, what is free, and how to check your balance — is covered in Compendium Chapter 2 (The ArcGIS Ecosystem). Her working rule: run every credit-consuming tool once on paper first, decide exactly what inputs it needs, then run it once for real.

Getting shops, candidates, and competitors onto a map

The first hands-on move was building three small layers: existing shops (two points), candidate sites (six points), and competitors. For the shops and candidates she typed a short spreadsheet — name, address, and a couple of attribute columns — and brought it into ArcGIS Online as a hosted feature layer, letting the platform geocode the addresses on the way in. The full menu of data-creation paths, including spreadsheet upload and sketching points directly in Map Viewer, is Compendium Chapter 11 (Creating Data); how hosted feature layers behave once published is Compendium Chapter 10 (Hosted Feature Layers).

Her reasoning for a hosted layer instead of a throwaway sketch: the candidate layer was going to accumulate attributes over the coming weeks — rent quoted, parking notes, scoring fields — and she wanted one durable table she could edit as facts came in, not a drawing she would redo. She spent ten minutes on the schema before publishing: a text field for the site name, numeric fields reserved for the scores she knew were coming, and a short domain-style set of allowed status values (candidate, shortlisted, rejected) so future-Marisol could not typo her way into a mess. That

instinct — design the fields before the data arrives — is the whole argument of Compendium Chapter 12 (Schema Design).

Competitors took a different path. Rather than buying business-listing data, she spent an evening building her own competitor spreadsheet from local knowledge, an online map search, and a drive around the two unfamiliar neighborhoods: every coffee-forward business she would consider a substitute, with a column distinguishing national chains from independents. The layer was small enough that geocoding it consumed a trivial number of credits. She accepted from the start that this layer was opinionated and incomplete — a hand-built competitive picture, not a census of the coffee market — and wrote that caveat directly into the layer's item page description so the map would carry its own disclaimer. Judging and documenting data quality this way is treated in Compendium Chapter 5 (Finding Data).

Drive-time trade areas, not circles

With points on the map, the tempting next move was a one-mile ring around each candidate. Marisol deliberately rejected it. Rivermont has a river through the middle and a limited-access highway on the east side; a circle pretends customers can travel through both. Her actual customers arrive by car and on foot along real streets, so she generated travel areas instead — drive-time polygons computed over the road network. In Map Viewer she opened `Analysis > Tools`, chose the travel-areas tool in the proximity group, and ran it against the candidates layer with two breaks: a short walk-scale time and a roughly ten-minute drive time. Then she ran the same tool, with the same settings, on her two existing shops. Proximity analysis — buffers, drive times, and when each is honest — is Compendium Chapter 16 (Proximity and Overlay).

Two reasoning points drove the settings. First, consistency: whatever time breaks she chose, every site had to get identical ones, or the comparison would be meaningless. Second, cost discipline: travel-area generation consumes credits per input feature, so she ran all eight sites (six candidates plus two existing shops) in as few tool runs as possible rather than experimenting one polygon at a time. When the polygons came back, the river did exactly what she expected — two candidates that looked equally positioned as the crow flies had visibly different drive-time footprints because only one had a bridge nearby.

Demographics from the Living Atlas, on a budget

Next she needed people inside those polygons. The premium route is the enrichment tool, which appends demographic variables to your features and charges credits per variable per feature. The budget route, which she took first, is the Living Atlas: Esri curates free, ready-made demographic layers — population, households, income, age structure — that anyone can add to a map. In Map Viewer she used `Layers > Add`, switched the search scope to Living Atlas, and pulled in a tract-level layer of current-year population and household income estimates. Finding, evaluating, and citing Living Atlas content is Compendium Chapter 5 (Finding Data).

To turn "a demographic layer under my polygons" into numbers per candidate, she used the summarize-within tool in `Analysis > Tools`, aggregating the demographic features inside each drive-time polygon. This is standard overlay work — Compendium Chapter 16 (Proximity and Overlay) again — and it produced a small result table: for each candidate, an illustrative population figure, household count, and income profile inside the ten-minute drive area. She reserved the paid enrichment tool for one final pass on the two eventual finalists only, where a richer variable set (daytime workers, spending on food away from home) justified the credit spend on exactly two features instead of six. Deciding which analysis tier each stage of a project deserves is exactly the kind of credit judgment Chapter 2 frames.

She also counted competitors per trade area the same way — summarizing her competitor layer within each candidate polygon, keeping chains and independents as separate counts, on the theory that a corner already anchored by two national chains is a different proposition than one with a single independent.

The misstep: ranking on raw totals

Here is where Marisol went wrong, and it is the most instructive moment in the project. With the summary table filled in, she ranked the six candidates by total population inside each drive-time area. Candidate D — a storefront in the Milldam district — won walking away, with an illustrative headline number nearly half again larger than the runner-up. She got as far as drafting the email to her broker naming D a finalist.

Then she toggled her existing shops' drive-time polygons back on and saw the problem: Candidate D's trade area overlapped heavily with the trade area of her original downtown shop. A large share of the "market" that made D look dominant was population Cardinal Coffee already served. Her ranking was answering "which site has the most people nearby?" — but her sticky-note question had been "which site has the strongest customer base *my existing shops do not already serve*?" The raw total silently rewarded cannibalization.

The recovery was mechanical once the error was visible. She used overlay tools — Compendium Chapter 16 (Proximity and Overlay) — to erase the existing shops' trade areas from each candidate's polygon, producing a "net new territory" polygon per candidate, then re-ran summarize-within against those clipped shapes. The re-scored table told a different story: Candidate D fell to the middle of the pack, while Candidate B, in the Northgate neighborhood across the river, rose to the top on net-new population despite a smaller gross number. The lesson she wrote in her project notes: every summary statistic answers a specific question, and the most dangerous analysis error is computing a correct answer to the wrong question. The gross-total map was not wrong; it was irrelevant.

A simple weighted score

With honest inputs, Marisol built the scoring model — deliberately simple, because a small business does not need a black box it cannot defend to a bank or a skeptical spouse. She chose four factors and illustrative weights that summed to 100: net-new population within the drive-time area (40), household income fit for her price point (25), competitor pressure, inverted so fewer competitors scores higher (20), and a walk-scale factor from the short travel-time polygon (15). For each factor she rescaled the six candidates' values to a 0–100 range, multiplied by the weights, and summed.

She did the arithmetic in two places on purpose. First in a spreadsheet, because a spreadsheet is auditable and easy to argue with — she could change a weight and watch the ranking move, which doubled as a sensitivity check (the top two candidates stayed on top under any reasonable weighting; the third and fourth traded places constantly, which told her the gap between them was noise). Then she wrote the same formula as an Arcade expression in the candidate layer's pop-up, so anyone opening the web map could click a site and see its score with the component values listed. Arcade expressions

from first principles are Compendium Chapter 8 (Arcade from Zero to Fluent); building pop-ups that explain themselves is Compendium Chapter 9 (Pop-ups, Fields, and Labels).

For the final map she styled the candidates by score using smart mapping defaults with light manual adjustment — graduated symbols, existing shops in a fixed distinct symbol, competitors small and gray so they read as context rather than subject. Styling choices that make a map argue clearly are Compendium Chapter 7 (Styling and Smart Mapping). Candidates B and E went to the broker as finalists, each with the map, the scoring table, and the caveats section below attached.

What this analysis cannot tell them

Marisol ended the project by writing down, honestly, what three weeks of browser GIS had not established — and this section did more for her credibility with her lender than the map did.

The demographics are residential. Census-derived layers count where people sleep, not where they buy coffee at 7:40 a.m. A site near an office cluster or a transit stop may massively outperform its residential numbers; only the finalists' enrichment pass touched daytime population, and even that is a modeled estimate. The drive-time polygons model typical travel, not her customers' actual behavior — no morning-rush directionality, no "which side of the street" effects, no knowledge of where people already stop on their commute. The competitor layer is her own judgment, current only as of the evening she built it. Nothing in the analysis knows rent, lease terms, build-out cost, parking reality, signage visibility, foot-traffic counts, or the quality of the landlord — several of which will dominate the final decision. And the demographic estimates themselves have vintages and margins of error; treating a modeled small-area estimate as a precise count is a category mistake (the habits for interrogating a dataset's lineage are in Compendium Chapter 5, and the statistical humility is in Compendium Chapter 17, Statistical and Pattern Analysis).

Her framing to the bank: the GIS work did not pick a site. It cheaply and defensibly eliminated four sites, so that expensive human diligence — site visits, traffic counts at dawn, lease negotiation — could concentrate on two.

The decisions that mattered

- **Writing the question before opening the map.** "Strongest customer base my shops do not already serve" dictated every layer, every tool, and ultimately exposed the cannibalization misstep; a vaguer goal would have let the raw-total ranking stand.
- **Drive times over circles.** Rivermont's river and highway made straight-line buffers a fiction; network-based travel areas cost some credits but changed which candidates looked reachable at all.
- **Living Atlas first, enrichment last.** Free curated demographic layers plus summarize-within answered most questions; the credit-consuming enrichment tool ran only on the two finalists, where richer variables were worth paying for.
- **Erasing existing trade areas before ranking.** The single recovery move of the project — scoring net-new territory instead of gross population — reordered the shortlist and was the difference between expanding the business and competing with herself.
- **Publishing the caveats with the map.** Naming what the analysis could not know converted the deliverable from a persuasive picture into a decision document a lender could trust, and pushed the remaining risk onto diligence steps that actually address it.

Case Study: Storm Outage Response at a Small Utility

A rural electric cooperative with a service territory spread across two counties decided, after one bad ice storm too many, to replace its paper-map-and-radio outage response with a GIS-based storm playbook: clean asset layers, a Field Maps workflow crews could run offline, an operations dashboard the dispatch room could trust, and a standing after-action review of the data itself. This case study walks through how they built it, in the order they built it, including the failure their first real storm exposed and the schema change that fixed it. The cooperative, its staff, and every event described here are fictional composites assembled to illustrate technique; no real organization or person is depicted, and all numbers are plainly illustrative.

The starting condition

Call the cooperative Miller Creek Electric. Its GIS footprint at the start was one part-time GIS coordinator, Dana, an ArcGIS Online organization with a handful of viewer accounts, and a set of shapefiles exported years earlier from a staking package: poles, transformers, line segments, and substations. During storms, dispatch worked from a laminated wall map and a whiteboard. Crews called in outage locations by landmark. Nobody could say, at any given moment, how many members were without power or which crew was closest to a given feeder.

The general manager gave Dana one directive after the last ice storm: "Next storm, I want to look at one screen and know where we stand." That sentence became the acceptance test for everything that followed.

Move 1: Rebuild the asset layers before touching any app

Dana's first decision was to resist the temptation to publish the old shapefiles as-is and start building apps. The reasoning: every downstream piece — field forms, dashboard counts, after-action queries — would inherit whatever the asset schema was, and the exported shapefiles had free-text fields, inconsistent pole numbering, and no domains at all. A "TRANSFORMER_TYPE" field contained a dozen spellings of the same three values. Fixing that after crews had collected a storm's worth of edits would be far more painful than fixing it first.

So the first month went to schema design, the discipline covered in Compendium Chapter 12 (Schema Design). Dana defined coded value domains for equipment type, phase, and voltage class; enforced a consistent facility ID pattern; and split the old catch-all "NOTES" column into structured fields where the notes actually encoded data (installation year, conductor size) while keeping a genuine free-text remarks field for everything else. The pole and transformer layers got a relationship in concept — every transformer references its pole's facility ID — so that a damaged pole could later be traced to affected equipment.

The cleaned data was loaded and published as hosted feature layers, the publishing mechanics and view strategy being the territory of Compendium Chapter 10 (Hosted Feature Layers). Dana published one editable outage-reporting layer and created read-only views of the asset layers for anything public-facing or dashboard-facing, so no app

ever pointed at the editable master with more permission than it needed. Editing itself was configured deliberately: editors could add and update but not delete, and editor tracking was turned on so every record carried who created and last edited it — a decision that paid off during the after-action review.

One layer was new rather than cleaned: an **outage incidents** point layer. This was the layer crews and dispatch would actually edit during a storm. Its first-draft schema had a status domain (Reported, Assessed, Crew Assigned, Restored), a cause domain (Tree, Equipment Failure, Ice Loading, Vehicle, Unknown), an estimated-members-affected integer, and a remarks field. Hold that first draft in mind; it is where the instructive failure lives.

Move 2: Basemap and territory data

Before building the field experience, Dana spent a short stretch assembling reference layers: county road centerlines from the state clearinghouse, parcels from the county, and a Living Atlas imagery basemap. The decision to pull authoritative external data rather than digitize roads in-house follows the sourcing judgment covered in Compendium Chapter 5 (Finding Data). The reasoning was operational: crews navigate by road name and member driveway, not by feeder ID, so the field map needed the same reference fabric the crews carry in their heads.

Move 3: The web map that everything else consumes

Dana authored a single "Storm Operations" web map in Map Viewer — asset layers styled for at-a-glance reading, outage incidents symbolized by status with unique values, and pop-ups configured so a tap on any incident showed status, cause, crew, and time reported without scrolling. Map Viewer mechanics are Compendium Chapter 6 (Map Viewer Complete Reference); styling choices lean on Compendium Chapter 7 (Styling and Smart Mapping), and the pop-up and field-formatting work is Compendium Chapter 9 (Pop-ups, Fields, and Labels).

The deliberate decision here was to make one map the shared source for both the field app and the dashboard, rather than authoring separate maps per app. The reasoning: during a storm nobody has time to reconcile two maps that drifted apart. One map, consumed everywhere, means one place to fix a symbology or pop-up problem.

A small piece of Arcade earned its keep in this map: an expression that computed hours-since-reported from the report timestamp and showed it in the pop-up, so dispatch could see aging incidents without doing mental arithmetic. Arcade from first principles is Compendium Chapter 8 (Arcade from Zero to Fluent).

Move 4: Field Maps, built for no signal

Miller Creek's territory has long stretches with no cellular coverage — exactly the places storms hit hardest. So the Field Maps design started from the assumption of disconnection, the workflow treated in depth in Compendium Chapter 30 (Field Maps, Survey123, QuickCapture).

Dana enabled the web map for offline use, defined map areas covering each of the co-op's four operating districts, and had crews download their district's area onto tablets during fair weather, as a standing checklist item rather than a storm-morning scramble. The form crews saw was ruthlessly trimmed: status, cause, members affected, photo attachment, remarks. Required fields were kept to the minimum a lineman wearing gloves in sleet would tolerate, because a form that is annoying in the field simply does not get filled in — the data quality battle is won at form design time, a theme Compendium Chapter 13 (Editing Workflows in Web and Pro) develops for both web and desktop editing.

Sync behavior was rehearsed, not assumed. Dana ran a tabletop exercise: two crews edited the same offline area, drove back into coverage, and synced. Watching how edits reconciled — and confirming that editor tracking attributed each record correctly — turned "offline should work" into "offline has been observed working."

Move 5: The dashboard dispatch actually watched

The operations dashboard, built with the techniques of Compendium Chapter 28 (Dashboards), consumed the same Storm Operations web map. Dana kept it to one screen with no scrolling, honoring the GM's one-screen directive: a map in the center, indicator cards for active incidents and estimated members affected, a serial chart of incidents by status, and a list of unassigned incidents sorted oldest-first. Every element filtered on the outage layer's status field, which is precisely why the status domain had been locked down in Move 1 — a dashboard can only count categories that exist as clean coded values, not as free-text spellings.

One restraint mattered: Dana refused to add elements for questions nobody in the dispatch room had actually asked. Each candidate widget had to answer "who looks at this during a storm, and what do they do differently because of it?" Most candidates failed that test and were cut.

The first storm, and the failure it exposed

The playbook's first live test came with a summer derecho. Mechanically, almost everything worked: offline areas synced when crews regained signal, the dashboard counted incidents in near real time, and the GM got the one screen he had asked for.

The failure surfaced about six hours in. Dispatch wanted to answer the question every co-op board member asks first: **which feeders are still out, and how many members hang off each one?** The dashboard could not answer it. The outage incidents layer had no feeder or circuit identifier — incidents were points in space, related to the electrical network only by proximity a human could eyeball, not by any attribute a dashboard could group on. The "members affected" number was a crew's windshield estimate typed into an integer field, not something derivable from the network. Dispatch spent the storm's second shift doing exactly what the project existed to eliminate: a person with a highlighter reconciling the screen against tribal knowledge of which taps feed which roads.

The recovery had two parts, one during the storm and one after. During the storm, Dana added a temporary workaround: a map-notes-style annotation the dispatch supervisor updated manually per feeder. Ugly, but honest — it kept the wall-map habit alive inside the new system rather than pretending the gap did not exist.

After the storm came the real fix: a schema change, executed the disciplined way rather than the panicked way. Dana added a `FEEDER_ID` field to the outage incidents layer with a coded value domain enumerating the co-op's feeders, added the same identifier to the line and transformer layers where it had never been carried over from the staking system, and made feeder a required choice on the Field Maps form — a single picker, cheap for the crew, transformative for dispatch. Because the change touched a production hosted layer with live data, Dana rehearsed it on a copy first and validated the domain assignments before touching the master, applying the data-quality habits of Compendium Chapter 14 (Data Quality). The lesson Dana wrote in the after-action file:

the schema was designed around what crews could report, not around what dispatch needed to ask. Both audiences have to be interviewed before the first field is typed.

Move 6: The after-action data review as a standing ritual

The last piece of the playbook was procedural, not technical: within a week of any major event, Dana runs a review of the outage data itself. Editor tracking answers who recorded what and when; a handful of attribute queries find incidents that were never closed out, records with Unknown cause that a follow-up call could resolve, and timestamps that reveal where reporting lagged reality. Simple summary statistics — restoration durations grouped by cause and feeder — come from the analysis patterns in Compendium Chapter 17 (Statistical and Pattern Analysis), and each review ends with at most one or two concrete changes to the schema, the form, or the dashboard. The discipline is that the review produces changes, not just a document; a playbook that does not absorb its own after-action findings decays back into the wall map.

By the second storm season, the derecho's feeder gap was a memory: the dashboard grouped active outages by feeder automatically, the temporary annotation layer was retired, and the highlighter stayed in the drawer.

The decisions that mattered

- **Schema before apps.** Cleaning fields, imposing domains, and standardizing IDs before publishing meant every downstream app inherited order instead of chaos — and the one gap that slipped through (feeder ID) proved how costly schema omissions are once live data exists.
- **One web map feeding both the field app and the dashboard.** A single authoritative map eliminated drift between what crews saw and what dispatch saw, and made every fix a one-place fix.
- **Designing Field Maps for disconnection first.** Pre-downloaded offline areas per district, rehearsed sync, and a gloves-in-sleet form kept data flowing from exactly the places storms cut off.
- **Interviewing dispatch, not just crews, about the schema.** The derecho failure came from modeling what the field could report rather than what operations needed

to query; the feeder-ID fix was one field, but it was the field the whole response turned on.

- **Making the after-action review produce changes, not documents.** Editor tracking plus a short list of standing queries turned each storm into schema and workflow improvements, which is what made the playbook durable instead of a one-project artifact.
