

KEEP CARING

A short guide to modernizing legacy systems.

blue tiger

A system becomes legacy the day you stop caring about the people who use it.

You can replace every line of code and still end up with a legacy system. Modernization is the act of caring again

HERE'S HOW TO START

① GET OUT OF THE BUILDING

INTERVIEW users and watch them work. Watching users navigate a system will reveal more about how the work is done than any interview can uncover. Let the interviews tell you what people are feeling and thinking.

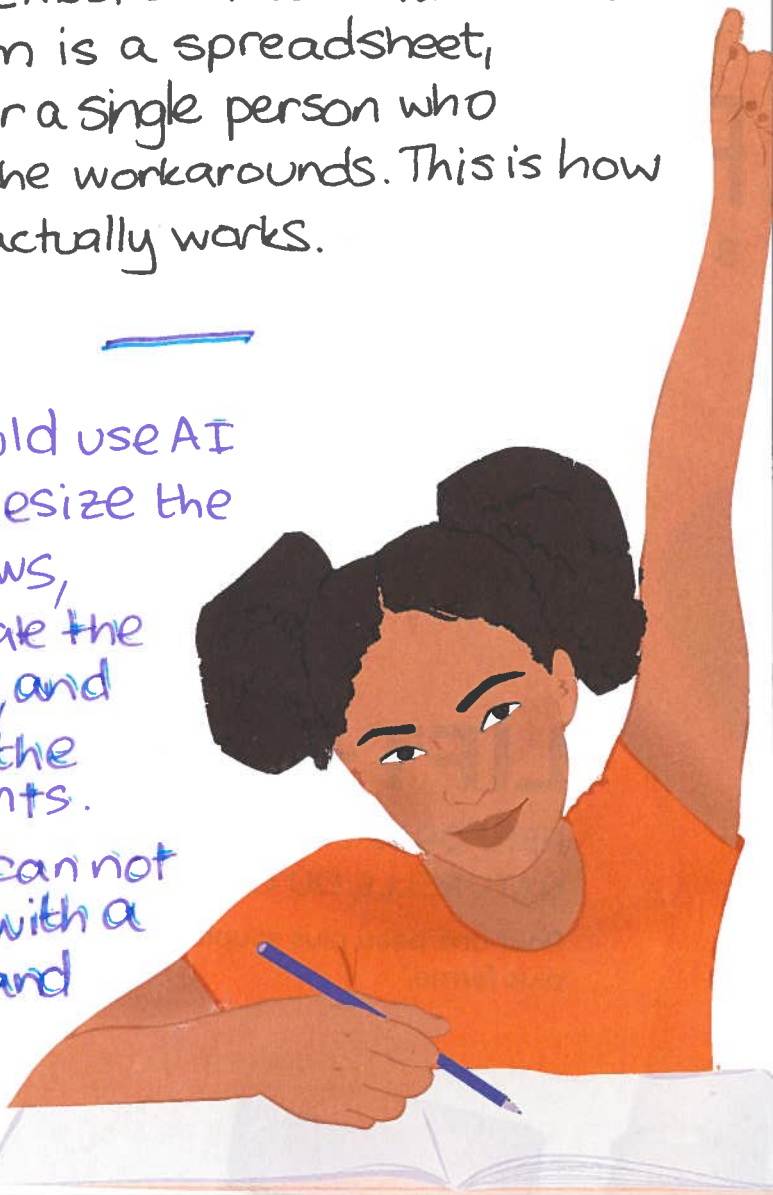
READ THE COMPLAINTS. Call center logs and issue trackers are a gold mine for information on how the system doesn't work. You'll learn about repeated patterns of neglect.

LOOK AT THE ANALYTICS. Analytics tell you where users get frustrated or confused, and if they are looking for more information.

FIND THE SPREADSHEET. Somewhere in a legacy system is a spreadsheet, group chat, or a single person who represents the workarounds. This is how the system actually works.

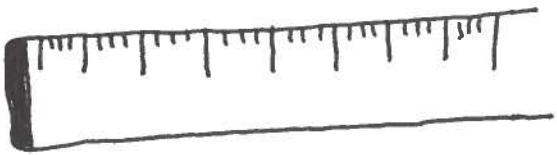
★ You should use AI to synthesize the interviews, investigate the call logs, and cluster the complaints.

However, AI can not take a ride with a caseworker and notice the sticky note on the dashboard.



(2) MEASURE WHAT MATTERS

How will you know whether your modernization worked? Start by identifying the one metric that matters the most: The single number that means it was all worth it. Then pick one or two more.



- Did the benefit reach the person who needed it?
  (applications approved)
- Did the applicant get an answer before the rent came due?
  (days to decision)
- Did people finish without needing support?
  (percent who complete without asking for help)

TEST EVERY DECISION AGAINST METRICS

If you're not improving outcomes as part of your modernization, you're just redecorating.



3 LOOK UNDER THE HOOD

Now that you know what's broken, its time to figure out why. Legacy systems are layers of policy decisions and technical shortcuts, each one reasonable at the time, none of them reasonable together.

READ THE INCIDENT LOGS. Where does the system fail, and how often?

MAP THE DEPENDENCIES BETWEEN SYSTEMS AND MODULES. Understand how data and decisions flow in and out of your system. The connections between things are the weakest points in a system.

FIND THE MANUAL STEPS. Anything that requires a human to do something in a specific order is a risk.

IDENTIFY THE DUPLICATED DATA. Every copy of data is a future discrepancy. And each discrepancy will eventually yield a frustrating outcome for the user.

SHE WAS HEADING

TOWARDS THE SUNSET

SHOUTING YEE HA ALL THE WAY.



GO ASK ALICE. Every legacy system has an Alice. The one person who knows the exact sequence of UNIX commands to run at 0200 and what happens if you don't. Alice is almost always close to retirement.

(4) NAME THE CAUSE

During synthesis, trace user pain points to their origin. This tends to be in one of two places.

(A) POLICY — The form is 20 pages long because it reflects the winding layered complexity of rules, regulations, exceptions, and special cases. Polishing the UX won't make meaningful improvements. Fix the policy.

(B) TECHNOLOGY — Your system was likely built for a different decade, a different volume of work, and you incurred tech debt to meet policy deadlines set by politicians in suits. Look for areas of fragility and unreliability. Just because a technology is old, it is not inherently flawed.

TALK TO THE POLICY EXPERTS. Many pain points are policy + technology tangled together. Skip the untangling, and you'll just reimplement the policy problem in new code. Policy owners often know what can be waived or simplified without an Act of Congress.

Policy Technology

TRACE IT BACK



USER PAIN

Fixing policy is slow and thankless work, but simpler policy will always yield better services.

5 Fix the RIGHT THINGS FIRST

Now you're ready to walk into prioritization with a list of specific problems to solve and outcomes to improve. Prioritization defines your modernization strategy, and you can't fix everything at once. Your goal should be to fix the right things first.

Plot each problem along two axes:

1. USER VALUE. How much does fixing this improve your outcomes? How many people are affected and how badly?

2. COMPLEXITY. How hard is this to change? How tangled is it with other systems? How well do you understand it?

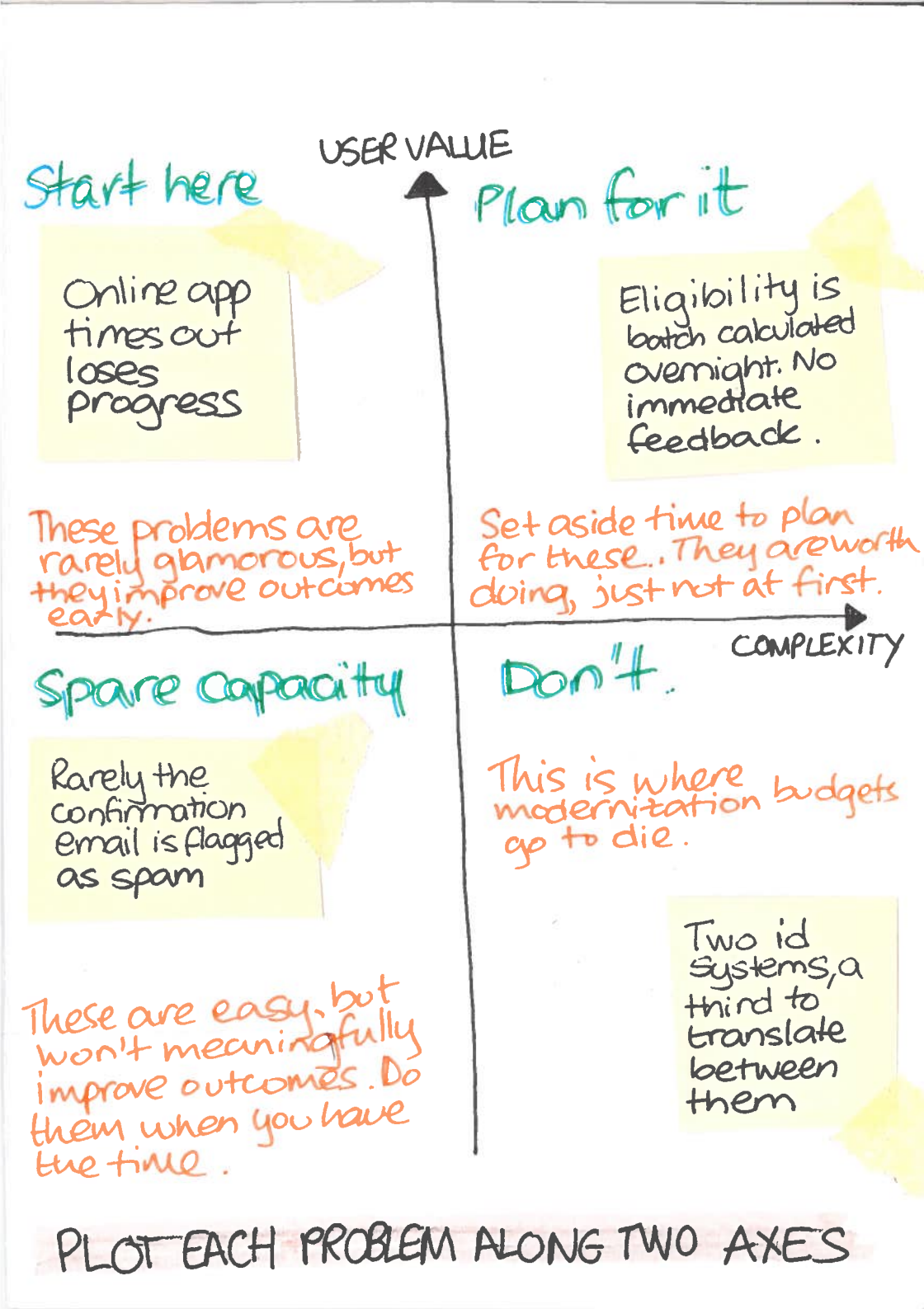
Four quadrants fall out of this (next page).

Resist two instincts:

a. Don't choose the hardest, most technically interesting problem because it is the one your engineers want to solve.

b. Don't choose the loudest problem. This is the one the suit keeps bringing up in meetings because it feels urgent.

Neither of these is the same as high value. Let the grid decide. EARLY WINS BUILD THE CREDIBILITY TO TACKLE HARDER PROBLEMS LATER.


USER VALUE
Start here
Online app times out loses progress
These problems are rarely glamorous, but they improve outcomes early.
Plan for it
Eligibility is batch calculated overnight. No immediate feedback.
Set aside time to plan for these. They are worth doing, just not at first.
COMPLEXITY
Spare capacity
Rarely the confirmation email is flagged as spam
These are easy, but won't meaningfully improve outcomes. Do them when you have the time.
Don't.
This is where modernization budgets go to die.
Two id systems, a third to translate between them
PLOT EACH PROBLEM ALONG TWO AXES

(6) KEEP CARING

Too many modernization efforts are treated as a single, one-time push to course-correct past decisions. But your system needs to continuously support reality. Users needs shift, and policies change.

Avoid these two common risks

1. Stalling: once you've landed a win or two (high-value, low complexity) attention moves elsewhere. Harder problems don't get solved.
2. Regression: Ignoring your users' needs can happen quietly. Needs can shift if you're not actively paying attention.

Fight both with the same habit: CONTINUOUS DISCOVERY.

* Keep talking to users.
* Report your metrics on a schedule.
* Assume new workarounds exist.
* Protect time for high value, high complexity work.

A system stays modern for exactly as long as the organization keeps caring for the people who use it.

That was true on day one. It'll be true 20 years later.

Writing and Layout
Shashank Khandelwal