



CRM Setup and Rebuild

Audit, build, integrate, scale.

CONTENTS

- 01 Audit, and what a week of reading turns up
- 02 Build, and the order it has to happen in
- 03 Integrate, and who owns which field
- 04 Scale, and what has to hold as you grow

Audit, and what a week of reading turns up.

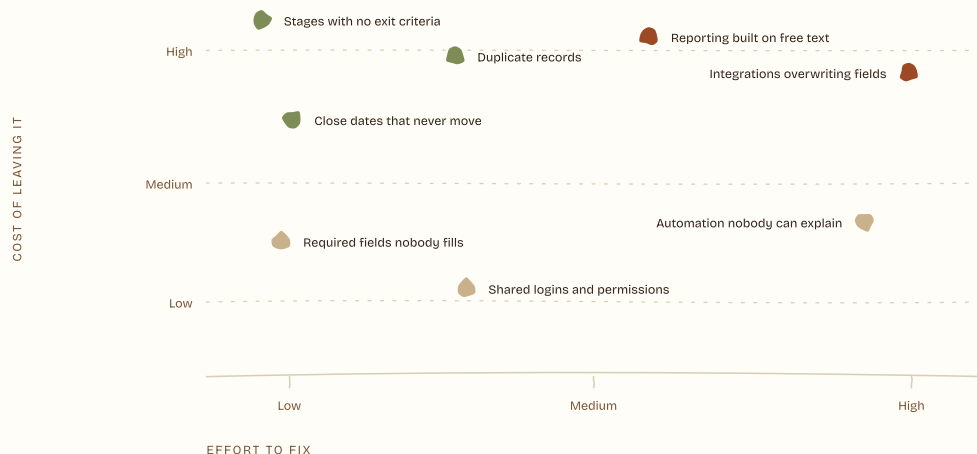
An audit is a week spent reading before anything gets touched. We go into the systems themselves rather than asking people what they believe is in there. The distance between those two answers is usually the most useful finding of the week.

Four things get read. **The data**, meaning how many records describe the same company, and how many fields carry something a query can actually use. **The process**, meaning what the stages are called and whether anything has to be true to move between them. **The adoption**, meaning who is entering data and who has quietly built a spreadsheet alongside it. **The reporting**, meaning which numbers leadership looks at, and whether the system can produce them without a person assembling it by hand.

What comes back is a written list of findings, each one placed against two axes: what it costs to leave alone, and what it costs to fix. Those two things correlate less than people expect, and that gap is the point of the exercise. The findings that do the most damage are usually cheap. The ones that feel most urgent in a meeting are often the expensive ones that can wait a quarter without hurting anybody.

We do not hand over a maturity model or a score out of ten. A number tells you nothing you can act on Monday morning.

A handful of findings turn up in nearly every system we read. Duplicate records, so no account has one history. Stages that describe what the rep is doing rather than what the buyer has committed to. Required fields that exist to satisfy a report nobody opens. Reporting built on free text, so one customer segment is spelled six ways. None of this reflects on the people who built it. Configuration drifts when the business changes faster than anyone has time to update it, which is every business.



Stages with no exit criteria Anyone can move a deal for any reason	Fix first Low effort	Reporting built on free text One segment spelled six ways	Fix next Medium
Duplicate accounts and contacts No record has a single history	Fix first Medium	Integrations overwriting fields Two systems taking turns on one value	Fix next High effort
Close dates that never move Every forecast built on a guess from March	Fix first Low effort	Required fields nobody fills Populated with a period to get past the screen	Later Low effort
Shared logins and permissions Everyone can edit a closed deal	Later Low effort	Automation nobody can explain Rules written by someone who has left	Later High effort

ARTIFACT 1

Findings from a typical audit, placed by what it costs to leave them alone against what it costs to fix them. The top left corner is where the first month of work goes.

Build, and the order it has to happen in.

Order matters more than any single decision inside it. Each layer rests on the one beneath, and changing a lower layer invalidates everything above it. That is why rebuilds which start at the dashboard get done twice.

Objects and fields first. What the business sells, who it sells to, and how those relate. This is where we settle whether a parent company and its sites are one record or several, and what a renewal is in your model rather than in a vendor's. Every later decision inherits from these answers.

Stages and exit criteria next. A stage is a claim about what the buyer has done, not a description of what the rep intends to do. Each one gets a written condition that has to be true before a deal moves. That single change does more for forecast accuracy than any scoring model you could buy.

Automation after that, and less of it than you would expect. Every rule is something a person will have to explain in a year, probably a person who was not there when it was written. We keep the count low and document what each one does and why it exists.

Reporting last, because a report is a question asked of the three layers underneath it. Build reports first and you hardcode assumptions about a model that has not settled yet.

Permissions and governance wrap all four. Who can create a field, who can edit a closed deal, and what happens to a pipeline when somebody leaves. This is dull, and it is the difference between a system that holds its shape and one that has four hundred fields by next summer.

We build in the system you already own. Nothing moves to a new platform unless the audit found something the current one genuinely cannot do, and that is rarer than the vendors would like it to be.

PERMISSIONS AND GOVERNANCE WRAP ALL FOUR

04

Reporting

Questions asked of the three layers below

03

Automation

The smallest number of rules that holds

02

Stages and exit criteria

What has to be true to move forward

01

Objects and fields

What you sell, and who you sell it to

BUILD ORDER

ARTIFACT 2

The build order. Each layer is a question asked of the one below it, which is why the sequence is not negotiable and why reporting is the last thing we touch rather than the first.

Integrate, and who owns which field.

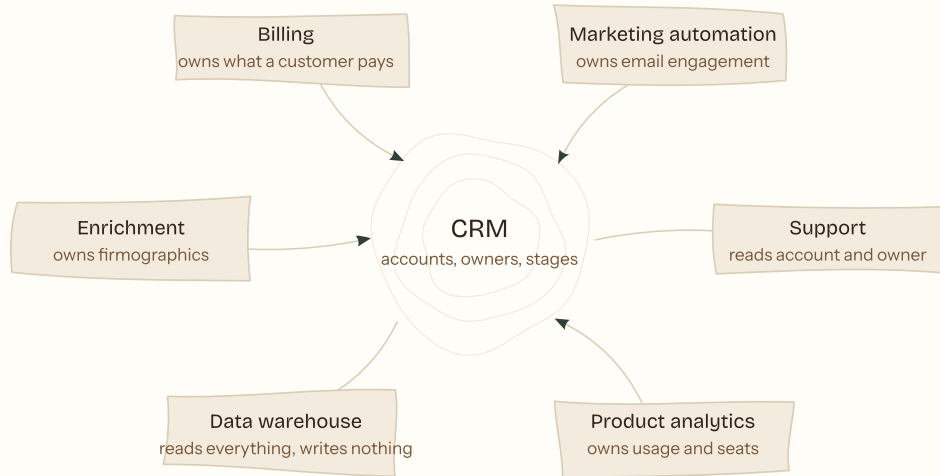
A CRM that disagrees with the other systems is a CRM people stop opening. Integration is mostly a question of ownership, and ownership gets decided before anything is connected.

For every field that exists in more than one place, one system owns it and the rest read it. Billing owns what a customer pays. The CRM owns who the account owner is and what stage the deal is in. Marketing automation owns email engagement. Product analytics owns usage. When two systems both write the same field they take turns overwriting each other, and the value on the record becomes a coin flip that depends on which job ran last.

Direction follows ownership. Most connections end up one way, which is cheaper to run, simpler to reason about, and far easier to debug at eight in the morning when a number looks wrong on a board deck.

Failures need somewhere to go. An integration that fails silently is worse than one that never ran at all, because everybody carries on using the data as though it arrived. Every connection we build reports its own failures somewhere a person will see them the same day.

And every connection gets written down. What it moves, in which direction, on what trigger, and who to call when it stops. Most of the integration problems we get asked to fix come down to documentation. Nobody could say what the connection was supposed to be doing in the first place, so nobody could tell it had stopped.



SOLID HEAD INTO THE CRM, THAT SYSTEM OWNS THE FIELD. HEAD OUTWARD, IT ONLY READS.

ARTIFACT 3

One field, one owner. Arrows point toward whichever system holds the authoritative copy. Four systems write into the CRM, two read out of it, and nothing writes the same field twice.

Scale, and what has to hold as you grow.

What works at twenty people gives way at eighty, and it rarely fails dramatically. Fields multiply, nobody owns the picklists, reports fork quietly, and one morning the number in the board deck stops matching the number in the system.

Three things decide whether a CRM survives growth.

Governance. One named person who approves new fields and stage changes. Without that, every new hire adds a field and nobody ever removes one. We have read systems carrying four hundred fields where fewer than sixty had been populated in the past year.

A written model. What a qualified opportunity is. What a renewal is. What counts as churn, and on which date. Growth means new people making these calls, and they will make them differently unless somebody has written the answer down where they can find it.

A quarterly review. One hour on what changed, what has gone unused, and what should be retired. It is the cheapest hour in the calendar and reliably the first one to get dropped.

UNDER 25 PEOPLE	25 TO 50	50 TO 100	100 TO 250
Habits	A process for changes	Written definitions	Active retirement
HAS TO EXIST One owner of the system. Stages with written exit criteria.	HAS TO EXIST A field request process, and somebody empowered to say no to one.	HAS TO EXIST Qualified, renewal and churn defined in writing, with dates.	HAS TO EXIST A quarterly review with the authority to retire fields and reports.
WHAT GIVES FIRST Nothing yet. The habits set here decide the next two years.	WHAT GIVES FIRST Field count doubles and the picklists fork into near duplicates.	WHAT GIVES FIRST Two teams report the same number and get two different answers.	WHAT GIVES FIRST The board number stops matching the system and nobody can say when it started.

Reviewed. Quarterly, one hour.

Owned. By one named person, not a committee.

Written. Or it did not happen.

ARTIFACT 4

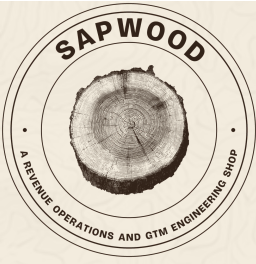
What has to be in place at each size, and what gives way first when it is not. The left column is cheap to do early and expensive to retrofit later.

Then the stack question.

Companies more often outgrow the shape of the stack around the CRM than the CRM itself. The pattern is familiar: a gap appears, a tool gets bought to close it, and two years later there are eleven licenses, four of them configured, and nobody can say which of them owns the customer record.

Before anything else gets added, it is worth knowing which of your current licenses are fully configured, which are half configured, and which are being paid for and never opened. That work usually pays for itself out of the renewals it cancels. It is a separate piece of work from a CRM build, and we treat it separately.

How we approach tech stack implementation



If any of this sounds like your system, we are easy to reach.

Thirty minutes, no charge, no deck. Tell us which platform you are on, roughly how many people touch it, and what stopped being trusted first. We will tell you on the call whether this is an audit, a build, or a smaller fix than either.

hello@sapwood.io

SAPWOOD

CRM SETUP AND REBUILD

SEPTEMBER 2026

DENVER, COLORADO