Doing the work is only half the job

Closing the gap between delivering the work and banking the money

Version 1.0 · ixlcore.com

Executive summary

Every business owner knows the satisfaction of delivering good work. Fewer talk about the quieter frustration that follows — the invoice that goes out a fortnight late, the job that was finished but somehow never billed, the cash that's stuck somewhere between "done" and "paid". Doing the work is only half the job. Getting paid for it is the other half — and it's the half that most software leaves you to manage by hand.

This paper argues that the gap between delivering work and banking the money is not a discipline problem. It's a systems problem. When the sale, the job, the stock and the invoice live in separate tools, revenue leaks through the seams — work delivered but never charged, invoices raised too slowly, cash tied up because no one can see the whole chain at once. The fix isn't more chasing. It's connection: when the quote, the job, the delivery and the invoice are the *same record* seen from different angles, work flows to cash automatically — nothing re-typed, nothing forgotten, and you get paid for everything you do, sooner.

The problem: the gap where revenue leaks

Order-to-cash is the full journey of a piece of work — from the moment you win it, through doing it and drawing down the stock or hours it consumes, to the invoice that finally lands in a customer's inbox and the payment that clears. In a healthy business, that journey is a single, unbroken chain.

In most growing businesses, it isn't. It's a relay race between systems that don't know each other exist. The sale happens in one place. The job is scheduled and done in another. The stock is tracked in a third. The invoice is raised in a fourth. Each hand-off is a person re-typing what the last system already knew — and every hand-off is a place for revenue to fall through.

The leakage is rarely dramatic. It's a slow drip:

- **Work delivered but never billed.** A job gets finished, the team moves on, and the invoice is never raised — because nobody connected "done" to "charge for it".
- **Invoices sent late.** The work ended weeks ago, but the paperwork sat in a queue. Every day of delay is a day longer before you're paid.
- **Under-billing.** Extra hours, extra materials, a variation agreed on site — captured in operations, but never carried through to the bill.
- **Cash you can't see.** Money is owed at every stage of the chain, but because no single system holds the whole picture, nobody can say how much is really out there or when it's due.

None of this shows up as a line on a statement. But it's paid every month — in revenue earned and never collected, and in cash that arrives later than it should.

Why the usual answers fall short

The instinct is to fix each stage with the best tool for that stage — a CRM for the pipeline, a scheduling app for the jobs, an accounting package for the invoices. Each is good at its job. None of them close the gap, because the gap isn't inside any one stage. It's *between* them.

A CRM knows you won the work. It doesn't know whether the work got done, what it cost, or whether it was ever billed. The sale is where its story ends — and where the money's story is only beginning.

An operations or scheduling tool knows the work got done. But "done" sits in one system while "invoice" sits in another, so someone has to notice, and carry it across, by hand. If they don't, the job is delivered and the bill never follows.

An accounting package knows how to raise an invoice — but only from what's typed into it. It has no line of sight into the job that earned the revenue, so it can't tell you what's been delivered and still needs billing. It waits to be told.

So the business bridges the gap with people and spreadsheets. Someone re-keys the invoice from the job sheet. Someone keeps a running list of who's been billed and who hasn't. Someone reconciles, at month end, what four systems each believe to be true. This is skilled work spent entirely on stopping revenue from leaking — value that produces nothing except the absence of a problem, and only as long as the person keeping the list doesn't miss a row.

A different idea: one connected chain

The alternative isn't a better invoicing tool or a stricter billing process. It's to stop treating order-to-cash as a series of separate systems to be bridged, and start treating it as one chain on a single source of truth.

On a connected platform, the quote, the job, the delivery and the invoice aren't four records in four systems that have to be kept in step. They're **one record, seen from different angles**. Sales sees the deal. Operations sees the job. The warehouse sees the stock it draws down. Finance sees the invoice and the ledger entry. But underneath, it's the same thread of truth — so when the work moves forward, every angle updates at once, and nothing has to be carried across because nothing was ever apart.

That single difference — one record instead of four — is what closes the gap. There's no hand-off to drop, because there's no hand-off. There's no re-keying, because there's nothing to re-key. And there's no list of who's been billed, because the system already knows.

How it works

On IXL CORE, the four functions that make up order-to-cash — **CRM**, **Operations**, **Supply Chain** and **Accounting** — are connected applications on a shared foundation, built around one customer record and one source of truth. Here's how a piece of work travels:

- **A sale in CRM becomes a job in Operations.** When a deal is won, the work it describes flows straight into operations as a job — same customer, same scope, same figures. No one re-enters the

order; it simply arrives as work to be done.

- **The job consumes stock in Supply Chain.** As the work draws down materials, that movement is recorded against real stock — so what was used on the job and what's left on the shelf stay in step, without a separate stock-take to reconcile them.
- **Job costing feeds the bill.** The hours and materials the job actually consumed are captured as it runs — so the invoice reflects what was really delivered, including the variations and extras that usually get lost.
- **Delivery posts an invoice and a ledger entry in Accounting.** When the work is done, the invoice is raised from the job itself — and the corresponding entry lands in the IFRS-aligned ledger at the same moment, with the right tax applied for the country you're operating in.

The result is a chain that runs itself. Because delivery and billing are joined at the source, **nothing delivered goes unbilled** — the system knows a job is done and that it hasn't yet been invoiced, and it won't quietly let that slip. Work flows to cash without anyone shepherding it across the gaps, because the gaps are gone.

What it looks like in practice

Take a single order through both worlds.

Before — on disconnected tools. A salesperson closes a deal in the CRM and emails operations the details. Operations types the job into a scheduling app and does the work. Along the way, extra materials get used and an hour or two gets added — noted on a job sheet. When it's finished, someone eventually re-keys an invoice into the accounting package, working from that job sheet, and hopes they've caught the extras. A separate spreadsheet gets a new row so the team can remember this one was billed. The invoice goes out — a couple of weeks after the work ended. At month end, the job records and the ledger are lined up by hand to make sure nothing was missed. Something usually was.

After — on one connected chain. The salesperson closes the deal. The job appears in operations already carrying the customer, the scope and the price. The team does the work; the extra materials and hours are captured against the job as it runs. The moment it's marked delivered, the invoice is raised from the job — extras included — and the ledger entry posts alongside it, tax and all. The invoice goes out the same day. There's no spreadsheet of who's been billed, because the system knows. There's nothing to reconcile at month end, because there was only ever one record.

Multiply that across every order, every month. The difference isn't one faster invoice. It's a business that charges for everything it does, the moment it does it.

The outcome

When order-to-cash runs as one connected chain, the change is not that billing gets a little tidier. It's that the leaks close and the cash speeds up.

- **You bill faster** — because the invoice is raised from the job the moment it's delivered, not days later from a re-keyed job sheet.
- **You bill for everything** — because the extras and variations captured in operations flow straight

through to the bill, and no finished job slips past unbilled.

- **Your cash flow gets healthier** — because work turns into invoices, and invoices into collectible cash, without sitting in the gaps between systems.
- **Month end gets quieter** — because there's no second version of the truth to reconcile, and no spreadsheet holding the business together.

Doing the work was always the part you were good at. The other half — getting paid for all of it, sooner — is the part that a connected system finally does with you, instead of leaving to you.

About IXL CORE

IXL CORE is the Business Operating System for growing businesses — one connected platform for customers, money, stock, people and work, built around a single source of truth and shaped for how businesses across Africa actually operate. Its CRM, Operations, Supply Chain and Accounting applications share one customer record and one ledger, so work flows from the first quote to the final payment without ever leaving the system. Learn more at ixlcore.com, or start at app.ixlcore.com.