

# The hidden cost of legacy cross-border payments

How to prepare for ISO 2022 without replacing your existing payment systems



# Executive summary

Banks finished their move to ISO 20022 in November 2025. Many of the companies that pay suppliers overseas did not move with them and still send payment files their bank has to fix before it can use them.

That cost is real but scattered. Banks charge to repair a non-standard file. Some reject it outright, which delays the supplier, can hold up a shipment, and leaves the payment exposed to currency movement. Payments that arrive without structured data get matched to invoices by hand.

Swift has moved the deadline for retiring MT101, but not the requirement. Address standards are still tightening, bank reporting moves to new formats by 2028, and the work will not be reduced by delaying it.

Closing the gap does not mean replacing your ERP or treasury system. Read on to find out how your existing systems can be enhanced to comply with the changing regulations.



# THE INDUSTRY MOVED TO ISO 20022. MANY CORPORATE SYSTEMS DID NOT.

## 2.1 What changed

ISO 20022 is a shared way of organizing the information sent with a payment request. The older formats packed everything into a few free-text boxes. ISO 20022 gives each item its own labelled field: who is paying, who is being paid, each part of their address, the invoices being settled, the reason for the payment. The receiving system can read each item directly instead of working out what sits inside a block of text.

The banking side moved quickly once the timetable was fixed. TARGET2 migrated in March 2023 and CHAPS in June 2023. Fedwire moved across in a single day on 14 July 2025. On 22 November 2025, banks stopped sending cross-border payments to each other in the old MT format, and by August 2026, more than 98 percent of payment messages crossing the Swift network were sent in ISO 20022.

### ISO 20022, THE ROAD AHEAD

#### DELIVERED • the migration the banking industry completed



#### SEPTEMBER 2026

#### AHEAD • the requirements that remain open



Figure 1. ISO 20022: The road ahead

At that level of adoption, continued use of a legacy format is an exception the market handles case by case rather than a position it is built to accommodate.

## 2.2 The gap that remains

Inside a finance team, the picture looks different. An Enterprise Resource Planning (ERP) or treasury system produces an MT101 request for transfer, or a positional text file, or a CSV written to a specification a bank issued several years ago. The bank receiving it stopped using that format months ago. Something has to close that gap, and today it is usually the bank fixing the file, a Swift fallback service, or somebody in accounts payable doing it by hand. The same gap runs the other way. Banks now produce cash management statements in ISO 20022, and a system that can only read the old MT940 cannot take them in.

### WHERE THE GAP SITS

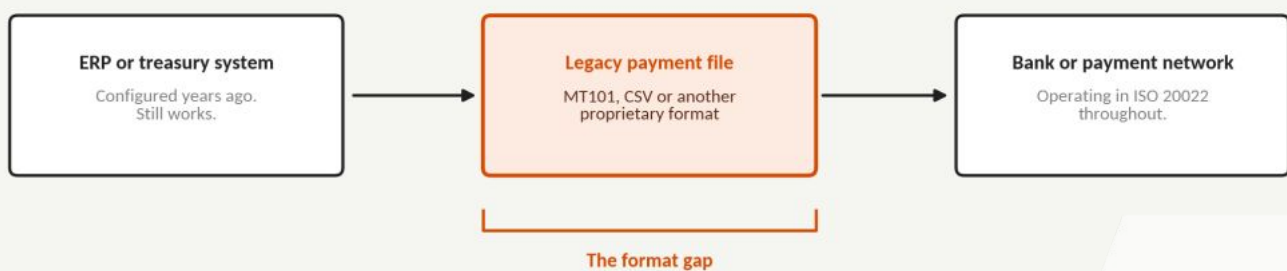


Figure 2. The format gap sits between systems that still work and a network that has moved on.

It is worth being precise about which gap this is, because the two halves of the migration are often discussed as one. Interbank cross-border messaging, meaning the pacs.008 and pacs.009 traffic that moves between banks, is the part that has closed. Corporate payment initiation is a separate track with its own timetable. What a company is exposed to today depends on which of the two it sits in, and the next section separates them.



## What happens if you are still sending legacy formats

Between banks, the change is already enforced. MT103 and MT202 no longer work for cross-border payments. Anything still sent in the old format is checked and converted by Swift before delivery, Swift charges the sender for doing it, and messages that fail the extra checks are rejected rather than converted.

Corporate payment initiation sits on its own track. Swift moved the date for retiring MT101 in August 2026 and is due to confirm revised timing by December 2026. That changes when the requirement bites, not whether it does. Swift continues to tell banks and market infrastructures to press ahead, and structured payments already move across the network every day.

Meanwhile, the cost of staying on an old format is already being paid, just not in one place. Every bank handles a non-standard file its own way. Some repair it and charge a fee. Some reject it and send it back. A company with five banking relationships is running five different sets of rules, and the rules keep changing.

Swift's fallback service is not a substitute for fixing this. In-Flow Translation will convert an MT101 into pain.001 before delivery and charge the sender for it, but it takes one message at a time rather than a file of them; it refuses any message carrying an unstructured address, and the message that comes out can be truncated, with names and addresses cut short to fit.



# WHAT LEGACY PAYMENT FORMATS COST THE BUSINESS

## 4.1 When a payment delay reaches the supply chain

Consider a payment to a component supplier, rejected because the beneficiary address was malformed. Finding the error, correcting it, and getting value to the supplier takes a week. By then the supplier is looking at an overdue invoice, and it holds the next shipment until the money arrives. Payment formats are not the only cause of late shipments. But when they are the cause, the cost shows up as lost sales rather than a bank fee.

## 4.2 Foreign-exchange exposure

That same week of delay has a second effect where the obligation is payable in a currency other than the payer's own, because the rate keeps moving between authorization and settlement.

Take a EUR 2,000,000 supplier payment authorized by a US company at a 1.0850 exchange rate. It is rejected, resubmitted, and settles a week later at 1.0980. The same obligation now costs about USD 26,000 more. The figures are illustrative, and the amount moves with the exposure and the length of the delay.

Nobody invoices that difference. In most management reporting, it lands in a Foreign Exchange (FX) variance line with no visible connection to a message format.



## 4.3 Reconciliation and visibility

The third effect outlasts the rejections, and it is the one finance teams most often underestimate. Structured data helps in both directions.

Going out, a pain.001 carries the invoice references for each transaction in their own fields, so a payment settling twenty invoices reaches the supplier with the twenty references attached rather than as one lump sum. The supplier can match it, which means fewer queries coming back.

Coming in, the bank sends Cash Management (camt) statements. A camt.053 is the end-of-day statement that replaces MT940 and MT950. A camt.052 gives the same picture during the day. A camt.054 is the advice for an individual credit or debit. All three carry structured detail on every entry, including the remittance information a customer sent with their payment. That is what lets incoming receipts be matched to open invoices automatically instead of by hand, and it is where most of the reconciliation effort actually sits.

Most banks can already send these formats. The constraint is usually at the company end, because an ERP or treasury system configured years ago reads MT940 and nothing else. When that happens, the bank falls back to the old format, and the extra detail is dropped before it ever arrives. The old statement formats retire by 2028, so this is a question of when rather than whether.

What a finance team actually notices is narrower than most benefit lists suggest. Fewer unallocated receipts at month-end, and a clearer answer when someone asks where a payment got to.

Establishing how much of this applies to a given company takes five questions, and none of them needs a consultant to answer.

### Five questions finance leaders should ask

1. What payment formats are our systems sending and receiving today? Include the ERP, the treasury system, and any other system that initiates payments or loads bank statements. The answer is often held by one person in the IT team and has never been written down.
2. How many cross-border payments were rejected, repaired, or delayed last quarter? If nobody can produce the number, that is itself useful information about how visible the cost currently is.
3. How is beneficiary information stored: in structured fields, or as free text? This determines how much work the structured address requirement will be when a date is set for it. The work sits in the vendor master, not in the payment file.
4. What have our banking partners told us they require? Ask each of them in writing for both the message and address formats. Requirements differ between correspondents, and they have been changing.
5. Who owns payment and beneficiary data quality? Treasury, accounts payable, procurement, or master data management. Where the answer is unclear, address data tends to degrade without anyone noticing.

# Modernize the payment message, not the finance stack

The conclusion most often drawn from all of this is the expensive one: that an ERP or treasury system producing a legacy payment format needs to be replaced or upgraded. It usually does not.

The format gap is a translation problem, and it can be solved at the point where the message leaves for the bank. A conversion layer sits in the payment flow, takes the file the existing system produces, and returns the ISO 20022 message the bank requires. The ERP carries on exporting what it exports. Approval workflows, user training, downstream integrations, and the reporting built on top of them all stay where they are. One step between the system and the bank is what changes. The same layer works on what comes back. Statements arriving from the bank as camt.053, camt.052 or camt.054 are converted into whatever the ERP or treasury system already reads, so the company gets the richer data without waiting for a system upgrade to consume it.

The difference against Swift's fallback service matters here. A conversion layer produces a proper ISO 20022 message, checked before it goes, in whatever volume the company sends. The fallback produces a translation of an old message, one at a time, and can cut data short to make it fit.

That is a much smaller job than a payments transformation program, and it is meant to be. It costs less and takes less time. What it changes for the business is set out below.



## What changes for the business

| What goes wrong today                                       | What changes                                                    | What the business gets                                                      |
|-------------------------------------------------------------|-----------------------------------------------------------------|-----------------------------------------------------------------------------|
| Payments fail on format or address errors                   | The file is checked and corrected before it reaches the bank    | Fewer failed payments and less rework in accounts payable                   |
| A supplier is paid late after a rejection                   | Errors surface before the money moves, not after                | Fewer held shipments and better standing with suppliers                     |
| One lump sum arrives against twenty invoices                | Invoice detail travels with the payment                         | Suppliers can match what you send, so fewer queries come back               |
| Statements arrive as MT940 and receipts are matched by hand | camt statements are converted into the format your system reads | Receipts match to open invoices automatically, and cash position is clearer |
| Nobody can say where a payment is                           | Status comes back to the system that sent it                    | Fewer calls to the bank and quicker answers internally                      |
| Each bank wants the file its own way                        | One layer handles each bank's requirements                      | Banks can be added or changed without touching the ERP                      |

### Who this is for

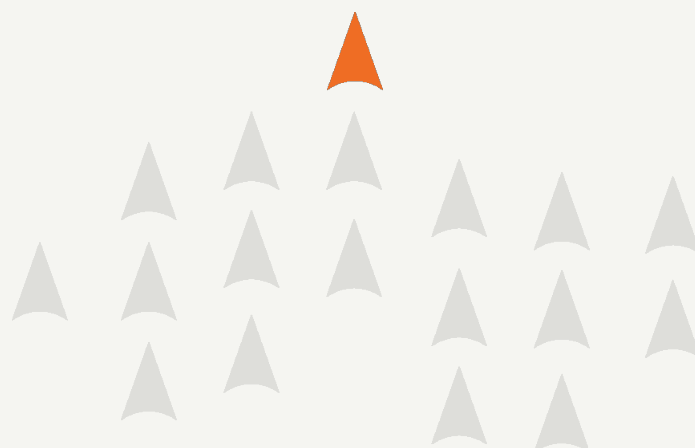
Companies that send cross-border payments by file upload or API from an ERP or treasury system; currently produce a legacy or proprietary payment format; have no in-house ISO 20022 expertise; and need to move quickly without disrupting existing payment operations.

# HOW THE CONVERSION WORKS, AND WHAT CHANGES FOR YOUR TEAM

## 6.1 The six steps

This is the one part of the paper that uses message-type terminology, because a finance reader evaluating the approach needs to see what the mechanism actually does.

- 1 Understand the existing format:** The engagement starts by establishing what the company's systems produce today and what the banking partner requires. In practice, this means sample files from the ERP or treasury system on one side, and the bank's current specification on the other.
- 2 Configure the conversion rules:** The plugin takes the applicable Swift translation guidance together with source and target message examples, and generates the validation and translation rules from them. Those rules are reviewable and can be changed before anything goes into use.
- 3 Validate the incoming message:** The file arriving from the source system is checked against the configured requirements before any conversion is attempted, so a defect in the source data is caught at the point it can still be traced back to a record.
- 4 Convert and validate the output:** An MT101 or proprietary payment initiation message will be converted into pain.001, with the required head.001 business application header generated alongside it. The output is validated before it leaves.
- 5 Return the converted message to the payment flow:** In production, this step happens automatically, with no operator involvement.
- 6 Run the same steps in reverse for statements:** Cash management statements arriving from the bank are validated, converted into the format the receiving system expects, and delivered into the existing reconciliation process.



HOW THE CONVERSION WORKS, IN BOTH DIRECTIONS

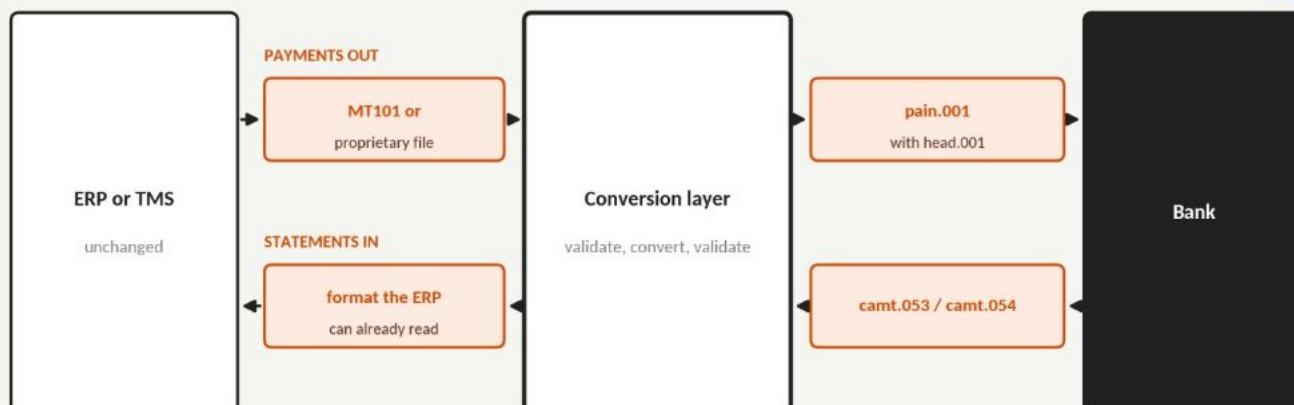


Figure 3. The conversion layer handles payments going out and statements coming back

## 6.2 What changes for the finance team

Four questions tend to come up first, usually from a controller.

**Do our systems have to change?** No. What must exist is a way to hand the outgoing file to the conversion layer and take the converted message back, which is usually a file drop or an API call the company already uses.

**What do our users do differently?** Nothing when submitting a payment. The visible change is in exception handling, because validation failures surface earlier and with a clearer reason attached to them.

**How do we know the output will be accepted?** Testing validates the output against the relevant CBPR+ requirements and against the receiving bank's own acceptance criteria, which are not always the same thing.



## 6.3 How to engage with Altimetrik

There are two routes in. A corporate client can engage directly, or be introduced by its bank. Both follow the same implementation process and arrive at the same result.

For a bank, the arithmetic is worth stating plainly. A single institution may have thousands of corporate clients still sending legacy formats, and each one is a recurring source of repair work, rejection handling, and client-servicing calls. Making an introduction carries no development cost and no implementation responsibility for the referring bank. This is the part of the paper written for client-servicing and transaction banking teams rather than for the corporate reader.

## 6.4 A leading global bank's payments modernization experience

A financial institution engaged Altimetrik to modernize legacy payment infrastructure, with three objectives: support for real-time transactions, improved reconciliation, and ISO 20022 readiness. Altimetrik assessed the existing architecture and implemented a modern payments hub spanning ACH, RTP, SWIFT, and FedNow.

Outcomes included improved operational efficiency, simpler integration across payment networks, better visibility over payment data, and an extensible foundation for further payment types.

# Why the industry is not ready, and what that means for you

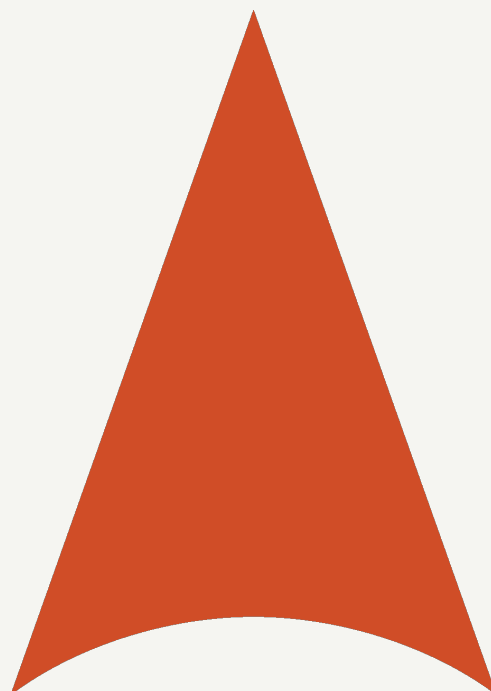
Getting the format right is only half the job. What goes inside the message matters just as much, and the requirement still to come is about the beneficiary address.

A structured address puts each part of an address in its own field: town, country, street, building number. Most companies hold all of it as one block of free text. Banks screen payments against sanctions and watch lists using that address data, and free text screens badly. It throws up false matches that stop good payments, and it hides real ones. Cleaner address data means fewer payments held for review and less manual checking, which is why the industry agreed to the change in the first place.

Swift extended the deadline in August 2026 because readiness was uneven across regions. Data from April 2026 showed around 61 percent of payments still carrying unstructured addresses. The extension was granted because the industry could not be ready, not because the requirement eased. The dates around it have not moved: Unstructured addresses retire in November 2027, and Fedwire runs its own release in the same month.

This is not a messaging project. Address data has to be correct in the systems that create the payment before any message format can carry it correctly. A supplier list holding 8,000 names whose addresses were typed into a single free-text box over fifteen years is the real scope of the task, and no conversion layer can supply a town name that was never recorded.

That work takes months, and it can start now. A company that begins collecting and structuring beneficiary data today will be ready whenever the cut-off date arrives. One that waits will do the same work against a shorter deadline, at the same time as everyone else, and competing for the same help.

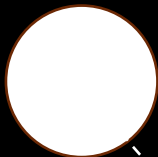


## Conclusion

The problem this paper describes is a finance problem rather than a standards problem. The messaging standard is settled, the banks have implemented it, and the remaining friction sits in the systems that originate payments and in the data those systems hold.

The five questions in section 4 are the place to start, and answering them costs nothing but time. Whatever those answers show, the work can be defined and priced on its own. It does not have to wait for a bigger IT project. Altimetrik runs scheduled ISO 20022 payment-readiness sessions for finance and treasury teams, working through those questions against the requirements now in force.

Book an ISO 20022 payment-readiness session [here](#) or reach out to us at [email@altimetrik.com](mailto:email@altimetrik.com).



## Building what's next.

Altimetrik runs scheduled ISO 20022 payment-readiness sessions for finance and treasury teams. Reach out to start with the five questions.

[aifirst@altimetrik.com](mailto:aifirst@altimetrik.com)

[www.altimetrik.com](http://www.altimetrik.com)