KEDBYTE

SOFTWARE · AI · AUTOMATION

HOW MONEY MOVES

Volume III — The Networks

The complete plain-English guide to payments and banking, followed by the engineer's version of the same thing.

WRITTEN BY

Shikhar Singh

Founder & Chairman

KedByte Technologies Private Limited

Five Volumes • Fifty-Four Chapters • First Edition

HOW MONEY MOVES

Volume III — The Networks

Author	Shikhar Singh
Designation	Founder & Chairman, KedByte Technologies Private Limited
Published by	KedByte Technologies Private Limited
Imprint	KedByte Press
Edition	First Edition
Subject	Payments, banking systems, financial technology
Extent	Five volumes, fifty-four chapters

Accuracy and currency. This book describes technology that changes. Specifications, prices, version numbers, product names and industry figures were checked at the time of writing and are dated in the text wherever they are liable to change. The chapters on artificial intelligence in particular describe a field that moves quickly; readers are told, in the text, where to re-check a figure before relying on it. Where experts disagree, the book says so rather than choosing a side.

Disclaimer. The information in this book is provided on an "as is" basis, for education. The security chapter is written for defence: it explains how classes of attack work so that systems can be built to resist them, and it deliberately contains no working exploit code. Neither the author nor KedByte Technologies Private Limited shall have any liability to any person or entity with respect to any loss or damage caused, or alleged to have been caused, directly or indirectly by the information contained in this book.

Worked examples. Parts F and G are built around one real network fault and one real version-control session, reproduced exactly as they were observed, including the addresses, the traceroute output and the error messages. Nothing in those traces has been altered or invented.

FOREWORD

From the Founder & Chairman

You tap a card against a machine and a shop lets you leave with the coffee. Somewhere behind that gesture, money you cannot see is supposed to move from a place you have never visited to another place you have never visited, and almost nobody involved can tell you how. Not the person who served you. Often not the person who installed the terminal. Frequently not the people who work at the bank.

That collective not-knowing is expensive. It produces businesses that cannot reconcile their own takings, engineers who store money in floating point, support teams who ask customers for the one reference number that cannot possibly help, and founders who discover the difference between a payment being authorised and a payment being settled on the day it costs them something. None of these are stupidity. They are the predictable result of an industry that documents itself for people who are already inside it.

This book is the outsider's way in, written to be useful to the insider. It begins with what money actually is, which is a record of obligation rather than a substance, and it ends with regulation, dispute and the ugly arithmetic of reconciliation. In between: the card in your pocket and the small computer embedded in it, the machine on the counter, the message formats that have carried the world's card traffic for forty years, the difference between a payment clearing and a payment settling, the rails that move money without cards at all, and what happens when any of it fails.

The method is the same one used in *How Computers Work*, and it matters more than the coverage. Every idea is explained twice. First in plain language, with an everyday comparison and a worked example, such that a twelve-year-old could follow it. Then again in the exact professional vocabulary, with the real field numbers, the real timings and the real institution names that a practitioner would use. Between the two sits a section that most writing on this subject omits entirely: an explicit statement of where the simple version stops being true. A useful simplification told without warning is how people end up confidently wrong, and in payments confidently wrong is a category of loss.

One thing about this book differs from its predecessor, and the reader should know it. How a transistor works will not change. Payments will. Scheme rulebooks are revised annually, regulatory caps move, and migrations between message standards run on live deadlines. The volumes are therefore arranged so that decay is contained: the mechanics that will outlive us sit in the early volumes, and the material that is true as of writing is gathered deliberately at the end, marked as

such. Where a figure is time-bound, this book says so rather than pretending to a permanence it does not have.

At KedByte Technologies Private Limited we build systems that move other people's money, which is a privilege that ought to come with an obligation to explain the machinery. If this book leaves one reader able to follow their own money from a card terminal to a bank ledger, and to say honestly which part of that journey they have verified and which part they have merely assumed, it has done its job.

Start at Chapter 1. It begins long before cards, and for good reason.

Shikhar Singh

Founder & Chairman

KedByte Technologies Private Limited

BEFORE YOU START

How to read this book

The shape of every section

Every section of every chapter is built from the same six blocks, always in the same order. Once you know the shape, you can navigate two thousand pages without ever being lost.

Block	What it is for
PLAIN in simple words	The idea for a complete beginner. No jargon at all.
PLAIN a picture in your head	One everyday comparison, and then a line telling you exactly where that comparison stops being true.
PLAIN a worked example	Concrete numbers, worked step by step.
PLAIN what is really happening inside	The mechanism, still in easy words, but with nothing hand-waved.
TECHNICAL the engineer's version	The correct professional vocabulary, exact units, real specifications, real figures, standard numbers and the commands that observe it.
WORDS remember these	Each term given twice: the plain meaning, then the technical meaning.

Two ways to read it

1. **The single pass.** Read everything, in order, from Chapter 1. This is the intended route and it works.
2. **The double pass.** Read only the blocks marked PLAIN, front to back. You will understand the whole of computing. Then return to the start and read the blocks marked TECHNICAL to be able to hold your own in an engineering conversation.

Conventions used throughout

1. Almost everything is written as short numbered lines, one idea per line, so nothing hides inside a paragraph.
2. *The honest version* marks the places where a simple explanation is not strictly true. The accurate model follows immediately.

3. The book states whether a thing is a **standard** (written down in a specification), a **convention** (everybody just does it this way), or an **implementation detail** (true of one product, not of the idea).
4. In the artificial intelligence chapters, claims are separated into established fact, active research, and marketing claim.
5. Anything that goes stale is dated. Check it again before you rely on it.
6. Every chapter ends with *Common wrong ideas* and a twenty-line summary. If you are short of time, read those two first.

Where to find things

1. **Chapter 1** is the map of the whole book.
2. **Chapter 52** is the reference desk: the numbers, the command cheat sheets and the two diagnostic decision trees.
3. **Chapter 53** is the master timeline and a glossary of more than seven hundred terms, each given in plain words and then technically.

Contents

VOLUME III

The Networks

The four-party model, ISO 8583 field by field, the several numbers all called the transaction reference, and where the money goes.

CHAPTER

1

The Four-Party Model

23.0 What this chapter gives you

1. You will be able to name the five roles in a card payment and say which of them holds a contract with which others, and which pairs have no contract at all.
2. You will be able to break a fifty pound sale into interchange, scheme fees and acquirer margin, and say who receives each slice.
3. You will be able to correct the commonest error in payments journalism, that a scheme charges shops 0.3 per cent, and explain why it is wrong twice over.
4. You will be able to explain why a merchant in a dispute has no cause of action against the issuer and must raise the matter through its own acquirer.
5. You will be able to say why a scheme with a three-party corporate structure can be classified as four-party, and what that reclassification costs it.
6. You will be able to tell an independent sales organisation from a payment facilitator by asking who holds the merchant contract and who bears the chargeback.
7. You will be able to explain why interchange is capped by statute in Europe and published by the scheme in the United States, and what Regulation II's exclusivity ban does to American debit economics.
8. You will be able to explain what IC++ pricing shows a merchant that blended pricing hides, and which of the three numbers is actually negotiable.
9. You will be able to explain why an outage at a card scheme stops new authorisations without destroying value that has already cleared.
10. You will be able to trace where the cost of a cashback reward finally falls, including on the customer paying cash.

The plain version

Imagine a very large school. Nobody carries cash, because cash gets lost. Instead every pupil has an account with one of several house banks, and every shop in the town around the school has an account with one of several other banks. The shops want to sell to the pupils. The pupils want to buy from the shops. The problem is that a shop banking with Barnaby's has no idea whether a pupil banking with Cresswell's actually has any money, and no reason to trust Cresswell's if it says so.

So the school sets up a switchboard in the middle, with a rulebook attached to it. The rules are the important part. They say what a shop is allowed to ask, what a bank must answer, how fast it must answer, what happens when the answer is wrong, and who pays when a pupil says "I never bought that." Every bank that wants to join signs the rulebook. Every shop signs a contract with its own bank, not with the switchboard and not with anybody else's bank.

Now watch what happens when a pupil buys a pair of trainers.

The pupil hands a card to the shop. The shop's till sends a message to the shop's own bank: "Someone is claiming to be a customer of another bank. This is the card number, this is the amount, this is my shop." The shop's bank looks at the first few digits of the card number, which act like a postcode, and sees that this card belongs to Cresswell's. The shop's bank does not phone Cresswell's. It passes the message to the switchboard. The switchboard passes it to Cresswell's. Cresswell's checks the balance, checks that the card is not reported stolen, decides, and sends one word back down the same chain: approved. The whole round trip takes about a second, and the shop's till prints a receipt.

That is four parties. The pupil. The shop. The shop's bank. The pupil's bank. And a fifth thing in the middle which is not a party to the purchase at all: the switchboard and its rulebook. In the real world the pupil is the *cardholder*, the shop is the *merchant*, the shop's bank is the *acquirer*, the pupil's bank is the *issuer*, and the thing in the middle is the *scheme*. Visa is a scheme. Mastercard is a scheme. Neither of them issued your card and neither of them has a contract with the shop.

Where the money actually goes

Here is the part that surprises people, so let us do it with real numbers.

Say the trainers cost fifty pounds, and the pupil pays with an ordinary British consumer credit card at a British shop. Your statement will say £50.00. The shop will not receive £50.00. It will receive something like £49.65.

The missing 35 pence splits three ways.

The largest slice, in this example fifteen pence, goes from the shop's bank to the pupil's bank. It is called *interchange*, and in the United Kingdom and the European Union it is capped by law at 0.3 per cent of the transaction for a consumer credit card and 0.2 per cent for a consumer debit card. Nought point three per cent of fifty pounds is fifteen pence. That money does not go to Visa or Mastercard. It goes to the bank that issued the card, and it is one of the reasons your bank is happy to give you a card, send you a replacement when you lose it, and refund you when someone in another country buys a television with your card number.

A smaller slice, call it five pence, goes to the scheme. These are *scheme fees*, and both banks pay them: the pupil's bank pays a fee for having its transaction switched, and the shop's bank pays one too. This is how Visa and Mastercard make money. It is a much thinner slice than most people assume.

What is left, about fifteen pence here, is the shop's bank's own margin. It is the price of the terminal, the fraud checks, the money the bank fronts to the shop tomorrow morning before it has actually been paid by the pupil's bank, and the risk that the shop goes bust next month owing refunds. The shop negotiated that margin when it signed its contract, and a big supermarket negotiates a very different number from a corner café.

Add them up: fifteen plus five plus fifteen is thirty-five pence on a fifty pound sale, which is 0.7 per cent. The whole deduction, taken together, is called the *merchant service charge*. The shop's bank quotes it, collects it, and passes the interchange and scheme fee parts onwards.

Approving is not paying

The second thing that surprises people is that nothing has moved yet. When Cresswell's said "approved", it did not send any money. It made a promise, and it set aside fifty pounds of the pupil's available balance so it could keep that promise. Actual money moves later, usually the next working day, and it moves in bulk. At the end of the day the switchboard adds up everything every bank

owes every other bank, cancels most of it out, and produces one number per bank: Cresswell's owes 4.2 million pounds today, Barnaby's is due 3.8 million. Only those net numbers are actually paid, through the ordinary banking system, bank to bank. Thousands of individual card payments become one transfer.

So a shop can be approved and still not be paid. If the pupil later says "that wasn't me", the rulebook lets the pupil's bank claw the money back from the shop's bank, and the shop's bank takes it back off the shop. That is a *chargeback*, and it travels back along exactly the same chain of contracts the payment came down.

Why the shop never phones your bank

Because it has no relationship with your bank whatsoever. There is no contract between them, no account, no phone number, no way to prove who is calling. The shop's only counterparty is its own bank. Your only counterparty is your own bank. The two banks' only shared counterparty is the scheme, and the scheme's whole job is to be the trusted stranger in the middle so that a card issued by a bank in Manila works in a bakery in Cardiff without those two institutions ever having heard of each other.

There is a practical reason as well as a legal one. Visa alone had close to five billion cards in circulation and more than 175 million merchant locations in its 2025 financial year. If every merchant had to arrange terms with every issuing bank, the number of contracts would be the number of merchants multiplied by the number of banks, and nobody would ever accept a foreign card. The scheme reduces that multiplication to addition: every bank signs one rulebook, every merchant signs one contract with one acquirer, and everything connects.

Now for the interesting variation. American Express, for most of its history, did not do this. It issued its own cards to its own customers and signed up its own shops directly, both ends, one company. There was no other bank in the picture, so there was no interchange, because there was nobody to pay it to. Amex simply decided what to charge the shop and kept all of it. That let it pay for very generous cardholder rewards, and it is why shops complained that Amex was expensive and why, for decades, plenty of them did not take it.

Where the plain version stops being true

The switchboard analogy makes the scheme sound neutral. It is not. A telephone exchange does not set the price of the call, decide who is allowed to be a customer, or fine you for connecting badly. A card scheme does all three. It writes the rulebook, and the rulebook runs to hundreds of pages of binding obligations on both banks. It sets the *default* interchange rate, which is the rate that applies whenever an issuer and an acquirer have not agreed something bilaterally, which in practice is nearly always. It operates the network, sells processing and fraud services in competition with its own members' vendors, arbitrates disputes between the two banks, and levies non-compliance assessments. Calling it a neutral intermediary is the single most misleading thing in the plain version. The European Union thought so too: Article 7 of the Interchange Fee Regulation forces schemes to separate the scheme business from the processing business, with independent accounting, organisation and decision-making, precisely because the two roles conflict.

Interchange does not go to the scheme, and scheme fees are not interchange. In the plain version I kept these separate, but almost every article you will read outside this book blurs them, and practitioners will judge you instantly for it. Interchange is defined in the Regulation as a fee paid between the issuer and the acquirer for each transaction. The scheme sets the default level but receives none

of it. Scheme fees are a completely separate charge, billed by the scheme to its licensed members, and they are not published. When a newspaper says "Visa charges shops 0.3 per cent", it is wrong twice over: Visa charges the shop nothing, and the 0.3 per cent goes to a bank Visa does not own.

"Four parties" undercounts the participants by a factor of two or three. A real card payment in 2026 touches, at minimum, a payment gateway, possibly a payment facilitator or an independent sales organisation, an acquirer processor that may not be the acquirer, the acquiring bank of record, the scheme's authorisation network, the issuer's processor, the issuing bank, and often a token service provider, a 3-D Secure directory server and access control server, and a digital wallet provider holding the credential. "Four-party" is a legal and commercial classification of who holds the licences and who bears the liability. It is not a headcount of the boxes the message passes through, and it never was.

Three-party and four-party are regulatory categories, not descriptions of plumbing, and a scheme can cross the line without changing its logo. American Express has not been purely three-party for a long time. Through Global Network Services it licenses other banks to issue Amex-branded cards, and through OptBlue it lets third-party acquirers sign small merchants for Amex acceptance and set their own price. Under Article 1(5) of the Interchange Fee Regulation, a three-party scheme that licenses other payment service providers to issue or acquire, or that issues with a co-branding partner or through an agent, "is considered to be a four party payment card scheme" for the purposes of the rules. The classification follows the licensing behaviour, not the corporate structure. That is a live commercial decision with a price attached, not a historical footnote.

Finally, the scheme does not move any money. It calculates. The instruction that actually transfers value between two banks is executed outside the card network, through settlement banks and national payment systems, and the card scheme's role ends with producing an authoritative net figure and a settlement report. A card network is a messaging and rule system that happens to determine who owes what. The payment leg is somebody else's infrastructure.

The technical version

The five roles and the contract lattice

The precise definitions are worth quoting, because they are legally operative in the European Economic Area and in the United Kingdom, where the Regulation was retained after withdrawal from the European Union. Regulation (EU) 2015/751, the Interchange Fee Regulation, defines them in Article 2.

An *acquirer*, Article 2(1), is "a payment service provider contracting with a payee to accept and process card-based payment transactions, which result in a transfer of funds to the payee". An *issuer*, Article 2(2), is "a payment service provider contracting to provide a payer with a payment instrument to initiate and process the payer's card-based payment transactions". A *payment card scheme*, Article 2(16), is "a single set of rules, practices, standards and/or implementation guidelines for the execution of card-based payment transactions and which is separated from any infrastructure or payment system that supports its operation". Note what that definition does: it makes the scheme a rulebook, not a network. The network is a separate thing which the same corporate group may or may not own.

A *four party payment card scheme*, Article 2(17), is "a payment card scheme in which card-based payment transactions are made from the payment account of a payer to the payment account of a payee through the intermediation of the scheme, an issuer (on the payer's side) and an acquirer (on the payee's side)". A *three party payment card scheme*, Article 2(18), is "a payment card scheme in which the scheme

itself provides acquiring and issuing services and card-based payment transactions are made from the payment account of a payer to the payment account of a payee within the scheme".

The commercial architecture follows from the contracts, and the contracts are worth setting out explicitly because almost every dispute in card payments is resolved by asking who has privity with whom.

Contract	Parties	What it governs
Cardholder agreement	Cardholder and issuer	Credit line or account access, liability for unauthorised use, fees, dispute rights
Merchant agreement	Merchant and acquirer	Pricing, settlement timing, chargeback liability, reserves, prohibited categories, termination
Issuer licence	Issuer and scheme	Right to issue branded credentials, BIN ranges, rulebook compliance, scheme fees
Acquirer licence	Acquirer and scheme	Right to sign merchants and submit transactions, settlement obligations, scheme fees
Scheme rules	Binding on both licensees	Message formats, timeframes, interchange defaults, dispute process, fines

There is no contract between the merchant and the issuer. There is no contract between the cardholder and the acquirer. There is no contract between the merchant and the scheme, though the scheme's rules reach the merchant indirectly because the acquirer is obliged to flow them down into the merchant agreement. This flow-down is the mechanism by which a merchant in Leeds ends up bound by a rule written in San Francisco without ever having signed anything from San Francisco.

What each party actually does

The *issuer* underwrites and funds. It performs know-your-customer checks, sets credit limits or account access, manufactures or provisions the credential, and holds the ledger against which authorisations are drawn. Operationally its critical function is the authorisation decision: within a scheme-mandated response window it must evaluate the incoming request against balance, velocity, fraud models and any Strong Customer Authentication requirement, and answer. It bears fraud losses on transactions where the liability rules place them with it, funds the interchange it receives, and handles cardholder disputes. It is also the party that answers to the cardholder in law: in the United Kingdom, section 75 of the Consumer Credit Act 1974 makes a credit card issuer jointly and severally liable with the merchant for breach of contract or misrepresentation on qualifying purchases, which is a liability the merchant's own bank does not carry.

The *acquirer* underwrites and funds in the other direction, and this is consistently underappreciated. When an acquirer signs a merchant, it is taking a credit exposure, not selling a service. If the merchant takes payment for holidays in January, collapses in March, and every customer charges back, the acquirer must return the money to the issuers and then try to recover from an insolvent estate. That is why merchant underwriting looks like lending, why acquirers impose rolling reserves and delayed settlement on high-risk categories such as travel and event ticketing, and why merchant category codes matter so much. The acquirer also allocates merchant identifiers and terminal identifiers,

registers its agents with the schemes, ensures PCI DSS compliance across its estate, and funds the merchant, typically on a T+1 basis, before it has itself received settlement.

The *scheme* is a rule-maker, a brand, a numbering authority, a specification body and, in most cases, a network operator. It allocates issuer identification numbers, publishes and versions the message specifications, operates the authorisation switch and clearing system, calculates settlement positions, publishes default interchange, arbitrates issuer-acquirer disputes, runs compliance and fraud-monitoring programmes with financial penalties, and licenses participants. Its revenue is scheme fees, which are charged on both sides and generally combine a per-transaction element with an ad valorem element and a set of behavioural fees, plus a growing book of value-added services.

The scale of that thin slice is worth making concrete. Mastercard reported gross dollar volume of $10.632 trillion for 2025 with net revenue of $32.8 billion. That is roughly thirty-one basis points of volume, and the figure is generous to the network, because net revenue includes cyber, intelligence and other value-added services that are not switching at all, and is stated after customer incentives and rebates. Visa's fiscal 2025, ended 30 September 2025, ran to $16.7 trillion of total volume and $14.2 trillion of payments volume, with 329 billion Visa-branded transactions of which 258 billion were processed by Visa itself.

The message flow

Card authorisation is carried in ISO 8583, a message format that has outlived several attempts to replace it. Every message begins with a four-digit message type indicator. Position one gives the version of the standard: 0 for the 1987 edition, 1 for 1993, 2 for 2003. Position two gives the message class: 1xx for authorisation, 2xx for financial, 3xx for file actions, 4xx for reversal and chargeback, 8xx for network management. Position three gives the function: 0 request, 1 request response, 2 advice, 3 advice response. Position four gives the origin: 0 acquirer, 2 issuer, 4 other.

That yields the message types a practitioner sees daily. An 0100 is an authorisation request. An 0200 is a financial request. An 0800 is a network management request, which is how the two ends of a link exchange sign-on, sign-off and echo tests to prove the connection is alive.

Immediately after the MTI comes the primary bitmap, sixty-four bits indicating which of data elements 1 to 64 are present. Bit 1 is not a data element; it signals the presence of a secondary bitmap covering elements 65 to 128. This is why ISO 8583 is compact: absent fields cost nothing, and a parser walks the bitmap rather than the message.

The elements that carry the commercial meaning of a four-party transaction are these.

Element	Contents	Notes
DE2	Primary account number	Variable length; the routing key
DE3	Processing code	Transaction type and account types
DE4	Amount, transaction	Minor units, no decimal point
DE7	Transmission date and time	Set by the sending institution
DE11	System trace audit number	Six digits, assigned by the originator
DE12	Local transaction time	Time at the point of service
DE13	Local transaction date	Date at the point of service
DE37	Retrieval reference number	Twelve characters, the reconciliation handle
DE38	Authorisation identification response	Six characters, the approval code

Element	Contents	Notes
DE39	Response code	Two characters; values defined by scheme rules
DE41	Card acceptor terminal identification	Eight characters
DE49	Currency code, transaction	ISO 4217 numeric
DE55	Integrated circuit card system related data	EMV chip data

DE11 and DE37 deserve a note, because they are what makes a four-party model auditable. The system trace audit number is assigned by the message originator and echoed back, which lets a party match a response to its own request across a link where messages may arrive out of order. The retrieval reference number is the handle by which the same transaction is identified later, in clearing, in a retrieval request, and in a chargeback, potentially months afterwards and by an institution that was not present at authorisation. Without a stable identifier travelling the full length of the chain, disputes between two banks that have never spoken would be unresolvable.

The critical structural point is that DE2 is the routing key. The acquirer's switch reads the leading digits of the primary account number, matches them against a table of issuer identification numbers, and determines which scheme owns the range and therefore where to send the message. Card numbering is governed by ISO/IEC 7812, which since its 2017 revision provides for eight-digit issuer identification numbers within a primary account number of up to nineteen digits, with a Luhn check digit in the final position. Visa ranges begin with 4; Mastercard uses 51 to 55 and the 2221 to 2720 range added in 2017; American Express uses 34 and 37 with a fifteen-digit account number. The acquirer therefore knows which scheme to route to and, from the fuller BIN table, which issuer and which product. What it does not have is any means of contacting that issuer directly.

Two processing models coexist. In the *dual-message* model, typical of credit, an 0100 authorisation request obtains a decision and a hold, and a separate clearing record submitted later, in a batch, actually presents the transaction for settlement. In the *single-message* model, typical of domestic debit and PIN-based networks, an 0200 financial request both authorises and clears in one exchange. The distinction matters commercially: in dual-message the authorised amount and the cleared amount can differ, which is what makes tipping, fuel pre-authorisation and partial shipment possible, and it is also what creates the reconciliation problem that DE37 exists to solve.

Both schemes are engaged in a long migration towards ISO 20022 message structures for card traffic, but ISO 8583 remains the working format in the overwhelming majority of acquiring estates, and any acquirer building today will still be writing bitmap parsers.

Clearing and settlement

Authorisation creates an obligation. Clearing records it. Settlement discharges it.

After authorisation, the acquirer submits presentments into the scheme's clearing system, historically Visa's BASE II and Mastercard's Global Clearing Management System with its Integrated Product Message format. The scheme edits each record against the rulebook, applies the correct interchange rate based on the transaction's characteristics, and produces reconciliation and settlement reports for each member. Interchange is not invoiced; it is applied as an adjustment inside the clearing figures, so the acquirer's settlement position is already net of the interchange it owes.

Settlement itself is a net obligation between each member and the scheme's settlement service, denominated in a settlement currency and discharged through nominated settlement banks over ordinary interbank payment infrastructure. The scheme is not a bank and does not hold customer funds in the general case. It computes positions and instructs. This is why an outage at a card scheme prevents new transactions from being authorised but does not, by itself, destroy value that has already cleared.

The fee stack, exactly

Three distinct charges sit between the price on the shelf and the money in the merchant's account, and confusing them is the most common analytical error in the industry.

Interchange is defined at Article 2(10) of the Regulation as "a fee paid for each transaction directly or indirectly (i.e. through a third party) between the issuer and the acquirer involved in a card-based payment transaction", and the definition explicitly adds that "the net compensation or other agreed remuneration is considered to be part of the interchange fee". It flows from acquirer to issuer.

Scheme fees flow from both licensees to the scheme. They are contractual, confidential, and revised on a published bulletin cycle.

Merchant service charge is defined at Article 2(12) as "a fee paid by the payee to the acquirer in relation to card-based payment transactions". It is the merchant's total cost and it contains the other two.

Interchange in the United States is unregulated for credit and published by the schemes. The Visa USA Interchange Reimbursement Fee schedule effective 18 April 2026 gives, among many hundreds of rates, the following.

Programme	Rate
CPS/Retail, Traditional Rewards credit	1.51% + $0.10
CPS/Card Not Present, Traditional Rewards credit	1.89% + $0.10
Visa Traditional, non-qualified consumer credit	3.15% + $0.10
CPS/Retail Debit, exempt issuer	0.80% + $0.15
CPS/Retail Debit, regulated issuer	0.05% + $0.21

The last two lines are the entire American debit interchange debate in miniature. Regulation II, at 12 CFR Part 235, caps the interchange an issuer may receive on an electronic debit transaction at the sum of 21 cents and 5 basis points of transaction value, with a further 1 cent fraud-prevention adjustment available to issuers that implement qualifying fraud policies. Section 235.5 exempts issuers that, together with affiliates, hold less than $10 billion in assets, which is why the schedule carries both an "exempt" and a "regulated" line for the same product. Section 235.7 separately prohibits network exclusivity: an issuer must enable at least two unaffiliated payment card networks on every debit card and may not restrict the merchant's routing choice. That single provision is what gives American merchants a genuine routing decision and what makes debit economics in the United States structurally different from credit. The Federal Reserve proposed in October 2023 to reduce the base component to 14.4 cents and the ad valorem component to 0.04 per cent; as of the end of 2025 the proposal had not been finalised, and banking trade associations were formally asking for its withdrawal.

In the European Economic Area and the United Kingdom the position is the opposite: interchange is capped by statute rather than published by the scheme. Article 3 caps consumer debit card interchange at 0.2 per cent of transaction value, with member state discretion for domestic debit including a per-transaction maximum of five euro cents combined with a rate not exceeding 0.2 per cent. Article 4

caps consumer credit card interchange at 0.3 per cent. Chapter II does not apply to commercial cards, to cash withdrawals, or to cards issued by three-party schemes, per Article 1(3).

Article 5 is the anti-avoidance provision and the one that catches designs that look clever on a whiteboard: any agreed remuneration with equivalent object or effect, including net compensation, is treated as part of the interchange fee. Article 2(11) defines net compensation as "the total net amount of payments, rebates or incentives received by an issuer from the payment card scheme, the acquirer or any other intermediary in relation to card-based payment transactions or related activities". An issuer incentive routed through the scheme rather than the acquirer is still interchange for the purposes of the cap.

The remaining business rules complete the picture. Article 6 prohibits territorial licensing restrictions, which is what makes cross-border acquiring within the Union lawful and is the legal basis for a merchant in one member state contracting with an acquirer in another. Article 8 protects co-badging and bars automatic mechanisms that force brand selection, while permitting the merchant to set a priority. Article 10 restricts the Honour All Cards Rule so that accepting one issuer's cards does not compel acceptance of all, save within the same brand and category of regulated consumer card. Article 11 preserves the merchant's right to steer. Article 12 requires the acquirer to give the payee itemised information showing the merchant service charge and the interchange separately, which is the legal foundation of what the market calls interchange-plus-plus, or IC++, pricing.

The practical consequence of Article 12 is a genuine change in merchant negotiating power. Under blended pricing, a merchant is quoted one rate for everything and cannot see whether a rise in its costs came from the scheme, the issuer mix, or its acquirer's margin. Under IC++, the invoice separates pass-through interchange, pass-through scheme fees, and the acquirer's own markup, and only the third number is negotiable. Larger merchants insist on IC++ for exactly this reason, and smaller merchants are usually on blended rates because they lack the volume to make the analysis worth anyone's time.

Cross-border interchange within Europe has been the live regulatory fight of the decade. Following the United Kingdom's withdrawal from the European Union, Mastercard and Visa raised interchange on UK-EEA card-not-present consumer transactions from 0.2 and 0.3 per cent to 1.15 and 1.5 per cent respectively, because the statutory caps applied to intra-EEA transactions that these had ceased to be. The Payment Systems Regulator's market review found that the two schemes were not subject to effective competitive constraint, that it could identify no documented justification for the increases, and that the cost to United Kingdom businesses ran to roughly £150 million to £200 million a year. The regulator consulted on a two-stage price cap, then in October 2025 abandoned the interim cap in the face of litigation about its powers and instead consulted on the methodology for a longer-term cap. Separately, the European Commission accepted binding commitments from both schemes in 2019 to cut inter-regional interchange for transactions in the EEA on cards issued elsewhere, reducing them by around 40 per cent on average.

Three-party schemes and what the difference buys

In a three-party scheme the issuing and acquiring functions sit inside one balance sheet. There is no interchange, because interchange is by definition a payment between two institutions and there is only one. The scheme sets the merchant discount rate unilaterally and keeps all of it.

American Express describes the resulting information position precisely in its own filings: "Wherever we manage both the card-issuing activities of the business and the acquiring relationship with merchants, there is a 'closed loop' in that we have direct access to information at both ends of the card transaction." It uses that position to underwrite risk, reduce fraud and target marketing, and it

describes its business as spend-centric, focused on driving spending on its cards rather than on lending balances.

Commercially, four consequences follow.

First, revenue concentration. A four-party scheme captures a thin switching margin, on the order of tens of basis points, while the fat part of the merchant's cost goes to thousands of issuers as interchange. A three-party scheme captures the whole discount rate. The Supreme Court, in *Ohio v. American Express Co.* in 2018, noted that Amex derives the bulk of its revenue from merchant fees, in contrast to networks whose issuers earn from lending.

Second, price setting. In a four-party model, the acquiring market is competitive and the acquirer's markup is squeezed by rivals, but the interchange floor is set collectively or by regulation and no individual merchant can negotiate it. In a three-party model there is no floor and no competitive acquiring layer at all: the merchant negotiates with the scheme itself, and large merchants therefore obtain real concessions while small ones do not.

Third, acceptance. A higher single price bought Amex a narrower merchant footprint for decades, which in turn constrained cardholder value. This is the classic two-sided platform bind, and it was the analytical heart of *Ohio v. American Express*. The Court, dividing five to four, held that Amex's anti-steering provisions, which prohibit merchants from dissuading customers from using Amex cards or promoting other cards more heavily, did not violate the Sherman Act, because the relevant market had to be defined to include both sides of the platform simultaneously. The opinion recorded 2013 shares of United States credit and charge card purchase volume of 45 per cent for Visa, 26.4 per cent for Amex, 23.3 per cent for Mastercard and 5.3 per cent for Discover.

Fourth, and most importantly for anyone designing a scheme today, the boundary is porous and crossing it has a regulatory price. Amex operates Global Network Services, under which partner institutions issue Amex-branded cards and in some markets also acquire, under independent operator arrangements, network card licence arrangements and joint ventures, with issuer rates agreed bilaterally with each partner. It operates OptBlue, under which third-party acquirers contract directly with small merchants for Amex acceptance and set the merchant's Amex rate themselves, exactly as they set the rate for the other brands. Amex's own merchant materials put the OptBlue eligibility ceiling at an estimated annual charge volume of less than $3 million, with certain categories such as charity, education, government, healthcare, insurance, residential rent and utilities exempt from the volume test.

Both of those programmes are, functionally, four-party arrangements wearing a three-party brand, and Article 1(5) of the Interchange Fee Regulation says so in terms. A scheme that licenses other payment service providers to issue or acquire, or issues through a co-branding partner or agent, is treated as a four-party scheme, which pulls it inside the interchange caps it would otherwise escape under Article 1(3)(c). Article 1(4) exempts three-party schemes from Article 7's scheme-processing separation requirement, so the classification carries structural obligations as well as pricing ones. The choice between three-party and four-party is therefore not a description of what a company is; it is a decision about how much distribution to buy and how much regulatory scope to accept in exchange.

The historical direction of travel is worth noting. Visa and Mastercard began as bank co-operatives: BankAmericard, launched in 1958 and licensed to other banks from 1966, became National BankAmericard Incorporated in 1970 and Visa in 1976; the Interbank Card Association, formed in 1966, became Master Charge and then MasterCard. Both were owned by their member banks, which is why the word "association" persists in older documents, and both demutualised in the 2000s, Mastercard

in 2006 and Visa in 2008, becoming ordinary listed companies whose customers are the banks that used to own them. The four-party model is a co-operative artefact that outlived the co-operative.

Distribution: acquirers, ISOs, payment facilitators and marketplaces

An acquirer licence is expensive to hold and slow to obtain, and acquiring banks are, as a class, poor at selling to small businesses. The industry therefore developed intermediaries, and the distinctions between them are precisely the distinctions of contract and liability set out earlier.

An *independent sales organisation*, or ISO, in Mastercard's vocabulary a member service provider, is a sales and servicing agent. It is registered with the scheme by a sponsoring acquirer, it may brand terminals and statements, and it earns residuals on the portfolio it introduces. What it does not do is hold the merchant contract or touch settlement funds. The merchant contracts with the acquirer. The ISO's role in law is agency, and the acquirer remains liable for its agent's conduct.

A *payment facilitator*, Visa's term, with Mastercard using the same concept and calling the merchant beneath it a sub-merchant, is a different animal entirely. The facilitator signs its own merchant agreement with the acquirer and then signs *sub-merchants* under that agreement, becoming, in Visa's vocabulary, the sponsor of *sponsored merchants*. It is the merchant of record. It receives settlement from the acquirer and disburses to its sub-merchants. It performs onboarding and underwriting under the scheme's standards, and it carries the liability for its sub-merchants' chargebacks and fraud, with the acquirer standing behind it. This is what allows an online platform to onboard a self-employed tradesman in ninety seconds when a bank's own onboarding would take three weeks.

Both schemes limit how far this can go by transaction volume. Mastercard raised its sub-merchant threshold from $100,000 to $1,000,000 in annual volume in 2014, and Visa followed with the same $1,000,000 threshold for sponsored merchants in the United States and Canada from 31 August 2017, granting merchants that crossed it a two-year window to complete a direct agreement with the acquiring bank. Above that level, the scheme's position is that the merchant is too large to be an anonymous line item inside somebody else's portfolio and must have a direct relationship with a licensed acquirer. The exact drafting, categories and effective dates are set out in the current editions of the Visa Core Rules and Visa Product and Service Rules and the Mastercard Rules, and any acquirer relying on them should read the edition in force rather than a summary, including this one: the schemes revise these provisions on a semi-annual bulletin cycle.

Marketplaces are a further variation, where the platform aggregates supply from many sellers but the commercial and legal position varies by jurisdiction and by scheme rule. The two questions that determine everything are who is the merchant of record and who bears the chargeback. A platform that answers "us" to both is operating as a facilitator whether or not it uses the word.

Finally, the acquirer of record is often not the entity doing the work. Much of the world's acquiring processing is done by specialist processors under contract to licensed acquiring banks, which is why a merchant may hold a contract with a bank it never speaks to, receive statements from a processor, be sold to by an ISO, and be onboarded by a facilitator, all inside what the rulebook regards as a single acquiring relationship.

Why the merchant never talks to your bank

Pull the threads together and the answer has four independent parts, any one of which would be sufficient on its own.

There is no contract. Privity runs merchant-to-acquirer and cardholder-to-issuer, and the only bridge is the pair of scheme licences. A merchant asserting a claim against an issuer has no cause of action

to assert it under; it must go through its acquirer, which raises the matter under the scheme's dispute rules, which is why chargeback representment is a process between two banks in which the merchant is a supplier of evidence rather than a party.

There is no settlement path. The merchant has a bank account at its acquirer or nominated for funding by its acquirer. It has no account at your bank and no means of receiving funds from it. Money reaches the merchant because the acquirer pays it, out of the acquirer's own funds, in advance of the acquirer being made whole through scheme settlement.

There is no addressability, and no trustworthy identity. The merchant knows the primary account number, from which it can derive the issuer identification number and therefore, via a BIN table, the issuing institution's name. That is a lookup, not a channel. There is no authenticated path from a terminal in a shop to an issuer's authorisation host that does not run through an acquirer's licensed connection and the scheme's switch, and there could not be: the whole security model depends on both ends being licensed, audited entities with cryptographic keys exchanged under scheme control.

And there is deliberate data minimisation. Under PCI DSS, the merchant is obliged to minimise what it stores, and network tokenisation is steadily removing the real account number from the merchant environment altogether, replacing it with a token that only the scheme's token service can map back to the underlying credential. The direction of travel is for the merchant to know *less* about the cardholder's bank over time, not more.

The nearest the two sides ever come is 3-D Secure, where an issuer's access control server briefly renders a challenge inside the merchant's checkout flow. Even that is mediated: the merchant's server talks to a 3DS server, which talks to the scheme's directory server, which routes to the issuer's access control server. The issuer and the merchant appear on the same screen and still never exchange a message directly.

What it costs, and who is actually paying

The final thing worth saying plainly is that the four-party model is a transfer mechanism as much as a payment mechanism, and every participant's economics depend on someone else's.

The merchant pays the merchant service charge, and prices it into the goods. The issuer receives interchange and spends most of it on rewards, fraud losses and the cost of running the credential. The scheme receives basis points from both banks. The acquirer receives a margin for taking credit risk it is often not thanked for taking. The cardholder pays nothing at the point of sale in most markets and receives a cashback or points benefit funded, ultimately, out of the merchant's margin and therefore out of the retail price, including the retail price paid by customers using cash.

That last transfer is the reason interchange is regulated in the European Union, the United Kingdom, Australia and a growing list of other jurisdictions, and the reason it remains largely unregulated for credit in the United States. It is a genuine policy disagreement about whether the price that balances a two-sided market should be set by the platform or by the state, and it is not going to be settled by an appeal to how the plumbing works. The plumbing works the same way either way. Only the numbers change.

23.98 Common wrong ideas

Wrong: interchange is what the card scheme charges. **Right:** interchange is paid by the acquirer to the issuer; the scheme sets the default rate, receives none of it, and earns its own money from separate and confidential scheme fees billed to both licensees.

Wrong: the scheme is a neutral switchboard. **Right:** it writes a binding rulebook, sets default interchange, arbitrates disputes between its own members, sells processing and fraud services in competition with its members' vendors, and levies non-compliance assessments.

Wrong: a four-party transaction involves four participants. **Right:** "four-party" is a legal and commercial classification of who holds the licences and bears the liability; a real payment also touches a gateway, processors on both sides, often a facilitator, a token service provider and 3-D Secure components.

Wrong: American Express is a three-party scheme. **Right:** through Global Network Services and OptBlue it licenses others to issue and to acquire, and Article 1(5) of the Interchange Fee Regulation therefore treats it as a four-party scheme for the purposes of the rules.

Wrong: the card scheme moves the money. **Right:** it calculates net positions and instructs; the transfer between banks is executed outside the card network through settlement banks and ordinary interbank payment systems.

Wrong: an approval means the merchant has been paid. **Right:** an approval is a promise and a hold on the cardholder's balance; settlement happens later and in bulk, and a chargeback can travel back down the same chain of contracts and take the money off the merchant again.

Wrong: a payment facilitator is just a reseller with better software. **Right:** it signs its own merchant agreement, becomes the merchant of record for its sub-merchants, receives settlement and disburses it, and carries their chargeback and fraud liability with the acquirer standing behind it.

Wrong: the acquirer is selling the merchant a service. **Right:** it is taking a credit exposure, funding the merchant on a T+1 basis before it has itself been settled, which is why merchant underwriting looks like lending and why travel and ticketing attract rolling reserves.

Wrong: 3-D Secure lets the merchant and the issuer talk to each other. **Right:** the merchant's server talks to a 3DS server, which talks to the scheme's directory server, which routes to the issuer's access control server; the two appear on the same screen and never exchange a message directly.

Wrong: the schemes have always been the listed companies they are today. **Right:** Visa and Mastercard began as bank co-operatives owned by their members and demutualised in 2008 and 2006, and the four-party model is a co-operative artefact that outlived the co-operative.

23.99 Chapter summary in 20 lines

1. A card payment connects a cardholder, a merchant, the merchant's acquirer and the cardholder's issuer, with a scheme in the middle that is not a party to the purchase at all.
2. The scheme exists so that every bank signs one rulebook and every merchant signs one contract, which turns a multiplication of relationships into an addition.
3. On a fifty pound British consumer credit card sale the shop receives about £49.65, and the missing thirty-five pence splits into interchange, scheme fees and the acquirer's own margin.
4. Interchange flows from acquirer to issuer, is capped at 0.3 per cent for consumer credit and 0.2 per cent for consumer debit in the United Kingdom and the European Union, and reaches the scheme not at all.
5. Scheme fees are a separate, confidential charge levied on both licensees, and they are a far thinner slice than most people assume.
6. The merchant service charge is the merchant's total cost and contains the other two, which is why the itemisation duty in Article 12 is the legal foundation of IC++ pricing.

7. The switchboard analogy fails because the scheme sets prices, writes binding rules, arbitrates disputes and fines its members, which is why Article 7 forces the scheme business to be separated from the processing business.
8. Privity runs cardholder-to-issuer and merchant-to-acquirer, bridged only by the two scheme licences, and almost every dispute in card payments is resolved by asking who has privity with whom.
9. The scheme's rules bind a merchant in Leeds only because the acquirer is obliged to flow them down into the merchant agreement.
10. The issuer underwrites and funds one side, makes the authorisation decision inside a scheme-mandated response window, and is the party that answers to the cardholder in law.
11. The acquirer underwrites and funds the other side, taking a credit exposure it is rarely thanked for, which is why reserves, delayed settlement and merchant category codes matter as much as they do.
12. Authorisation is carried in ISO 8583, whose four-digit message type indicator encodes version, class, function and origin, and whose bitmap makes absent fields free.
13. DE 2 is the routing key, DE 11 matches a response to its request across one link, and DE 37 is the handle by which the same transaction is found months later in a chargeback.
14. Dual-message and single-message models coexist, and the gap between authorised and cleared amounts in the dual-message model is what makes tipping, fuel pre-authorisation and partial shipment possible.
15. Authorisation creates an obligation, clearing records it, and settlement discharges it through nominated settlement banks over infrastructure the scheme does not own.
16. In the United States credit interchange is unregulated and published by the schemes, while Regulation II caps debit interchange and bans network exclusivity, which is what gives American merchants a genuine routing decision.
17. In Europe and the United Kingdom interchange is capped by statute instead, with Article 5 catching any equivalent remuneration, including issuer incentives routed through the scheme.
18. A three-party scheme captures the entire discount rate because there is no second institution to pay interchange to, and it historically paid for that with a narrower merchant footprint.
19. The boundary between three and four party is porous: licensing others to issue or acquire reclassifies a scheme, making the choice a decision about how much distribution to buy and how much regulatory scope to accept in exchange.
20. The four-party model is a transfer mechanism as much as a payment mechanism, funding cardholder rewards out of the merchant's margin and therefore out of the retail price, including the price paid by customers using cash.

Sources: Regulation (EU) 2015/751 (Interchange Fee Regulation) via EUR-Lex; 12 CFR Part 235 (Regulation II) via eCFR and the Federal Reserve; Visa USA Interchange Reimbursement Fees schedule effective 18 April 2026; Visa Core Rules and Visa Product and Service Rules; Visa Inc. Fiscal 2025 Annual Report; Mastercard Incorporated fourth quarter and full year 2025 earnings release; American Express Company Form 10-K and OptBlue merchant materials; Ohio v. American Express Co., 585 U.S. 529 (2018); UK Payment Systems Regulator market review MR22/2 into cross-border interchange fees; European Commission antitrust commitments decision IP/19/2311; ISO 8583 and ISO/IEC 7812 message and numbering standards.

2

CHAPTER

ISO 8583

24.0 What this chapter gives you

1. You will be able to explain what the bitmap does, and read the shape of a transaction off eight bytes of hexadecimal before parsing a single value.
2. You will be able to decode a four-digit message type indicator into version, class, function and origin, and say what `0100`, `0420`, `0800` and `1240` each mean.
3. You will be able to explain why the twelve digits `000000004250` mean nothing until you have read DE 49, and why the same digits are four thousand two hundred and fifty yen in Tokyo.
4. You will be able to say why "ISO 8583 compliant" on a vendor's data sheet is worth almost nothing, and what three questions to ask instead.
5. You will be able to explain why the 2023 edition reused version digit 2, and why that is a standards body formally conceding that the wire format is frozen.
6. You will be able to distinguish a request from an advice from a repeat, and say which of them may be refused and which may only be acknowledged.
7. You will be able to explain why the industry stayed on the 1987 edition even though the 1993 response codes are better in every respect.
8. You will be able to explain why a malformed bitmap produces a £4,250,000 purchase rather than a parse error at the malformed byte.
9. You will be able to say what DE 90 does, and why a reversal needs five separate sub-elements to name the message it wants undone.
10. You will be able to explain why what leaves a terminal is not an ISO 8583 message, and name the layers wrapped around it.

There is a sentence that gets repeated at payments conferences until it stops meaning anything: *ISO 8583 is the language of card payments*. It is not a language. It is closer to a filing convention — an agreement about which numbered box holds the amount, which holds the card number, and how to tell the person at the other end which boxes you bothered to fill in.

That is a smaller idea than "language", and it is the reason the format has survived. Languages evolve and fragment. Filing conventions calcify, and calcification is what you want in a system where a terminal made in 2003 has to be understood by a mainframe replaced in 2019, which has to be understood by a card issuer in a country neither has heard of. The format was first published in 1987, and the overwhelming majority of card authorisations on earth are still carried in messages whose first digit says *1987*. This chapter explains what is in one of those messages, digit by digit and field by field, and why the phrase "ISO 8583 compliant" on a vendor's data sheet tells you almost nothing you needed to know.

The plain version

Imagine you and a friend need to send each other messages by telegram, and the telegram office charges you by the letter. You will be sending the same *kind* of message hundreds of times a day: how much, whose account, which shop, what time. Writing out "Amount: forty-two pounds fifty" every time would bankrupt you.

So you agree on a form. Not a form you send — a form you both memorise. Box 2 is always the card number. Box 3 is always what kind of transaction this is. Box 4 is always the amount. Box 11 is always a serial number so you can tell one message from another. Box 39 is always the answer: yes or no.

Now you never have to write a label again. You only write the contents.

But there is a problem. Most messages do not fill in every box. A shop till sending a payment does not need Box 36, which holds data from a bank card's third magnetic track that nobody has used since the 1980s. If you leave boxes out, how does your friend know that the "000000004250" they are reading is the amount and not something else?

So you add one more thing at the very front of the message: a checklist. Sixty-four ticks in a row, one for each box, in order. A tick means "this box is in the message". A blank means "skip it". Your friend reads the checklist first, learns that boxes 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 37, 41, 42, 43 and 49 are coming, and then reads the contents of exactly those boxes, in exactly that order, and knows what each one is without a single label being sent.

That checklist is called a *bitmap*, and it is the whole trick. Everything else in ISO 8583 is bookkeeping around it.

A worked example

You buy a book. £42.50, at a bookshop in London, on 17 March 2026, at 32 minutes and 5 seconds past two in the afternoon. You tap your card. The terminal fills in the form:

Box	What it holds	What it says
2	Card number	4111111111111881
3	What kind of transaction	000000 (a purchase)
4	Amount	000000004250
7	Date and time sent	0317143205
11	Serial number	004417
12	Local time	143205
13	Local date	0317
14	Card expiry	2803
18	What kind of shop	5942 (a bookshop)
22	How the card was read	071 (tapped)
25	Circumstances	00 (nothing unusual)
32	Which bank is asking	454311
37	Reference number	607614430041
41	Which till	TERM0042
42	Which merchant	KEDBYTEBOOKS01
43	Merchant name and town	KEDBYTE BOOKS LTD LONDON GB
49	Currency	826 (pounds sterling)

Two things in that table are worth stopping on. The amount has no decimal point and no pound sign. It is twelve digits: `000000004250`. The decimal point is not sent, because Box 49 already said "826",

which means pounds, and everybody has agreed that pounds have two digits after the point. Four thousand two hundred and fifty pennies. Had the currency been Japanese yen, the same digits would have meant four thousand two hundred and fifty yen, and there would have been no decimal point to imagine. And the card expiry is `2803`, not `March 2028`: year first, then month, so the digits sort in date order if you line them up.

Somewhere between one and two seconds later, a message comes back the other way with almost nothing in it. The serial number, so the till knows which purchase this is about. A six-character code the bank made up, `A4T29B`, so that if there is ever an argument the two sides have something to point at. And Box 39, which is two characters long:

```
00
```

That is it. Two characters. `00` means yes. Anything else means some flavour of no, and the till has a printed list telling it what to say to the customer for each one. The receipt prints. You take your book.

Why it is not written out in words

Suppose you wrote that same purchase out with labels, the way a modern computer system usually would, so that every value has its name written next to it. It comes to 639 characters. Trim the labels down to bare numbers and it still comes to 289. Written in the numbered-box form with the checklist at the front, it is 181 characters, and if you use the compact way of writing digits — two squeezed into the space of one letter — it is 129. In 1987, a shop's card terminal talked to its bank down a telephone line at 1,200 bits per second, which is about 120 characters every second. So:

Form	Size	Time to send at 1,200 bits per second
Written out with full labels	639 characters	5.3 seconds
Labels shortened to numbers	289 characters	2.4 seconds
Numbered boxes, plain digits	181 characters	1.5 seconds
Numbered boxes, packed digits	129 characters	1.1 seconds

And that is one direction. Double it for the reply. The difference between the first row and the last row is the difference between a customer waiting eleven seconds at the till and a customer waiting two. The format was not designed to be elegant. It was designed to fit down a copper wire while somebody stood there holding their shopping.

Four kinds of message

There are only four things any of these messages can be. A **question** asks: may I take £42.50 from this card? An **answer** comes back the other way carrying the yes or the no. A **notice** says: I have already done something, and I am telling you about it — a shop whose terminal lost its phone line and took the payment on paper sends a notice, not a question, and the bank cannot refuse something that has already happened. A **cancellation** says: forget the last one. If the till asked for £42.50 and the answer got lost on the way back, the till does not know whether the money was set aside, so it sends a cancellation quoting the exact details of the message it wants undone, and keeps sending it until somebody confirms.

Question, answer, notice, cancellation. Everything else — sign-on messages, key exchanges, the nightly totals two banks send each other to check they agree — is one of those four wearing a different hat.

Where the plain version stops being true

It is not one form. It is an agreement to number the boxes. This is the most important correction in the chapter and the one the industry is loosest about. ISO 8583 specifies the shape of the envelope: a message type at the front, one or two checklists, then the contents in checklist order. It does not, in the current edition, even contain the list of what the boxes mean. In the third edition, published in July 2023, the message definitions, field definitions and code values were deliberately moved *out* of the standard and into a separately maintained register, so that they can be changed without republishing an International Standard. Two systems can both be entirely faithful to ISO 8583 and be unable to exchange a single message, because faithfulness to ISO 8583 does not settle what goes in box 48, or box 62, or what response code `05` means to you as opposed to me.

"Binary format" is only about a tenth true. The checklist genuinely is binary — eight raw bytes whose individual bits carry meaning. But the rest of the message is usually characters: digits and ordinary letters, written in ASCII or, on older mainframes, in EBCDIC. Some implementations pack numeric fields two digits to a byte and some do not, and the standard permits both, so the same logical message can be 129 bytes on one link and 181 on another. Sizes quoted for ISO 8583 messages are always sizes for a *particular implementation*, never for the format.

The box numbers do not mean the same thing everywhere. In the 1987 edition, data element 24 holds a network routing identifier; in the 1993 edition the same number holds a function code doing an entirely different job. Response codes are two characters long in 1987 and three in 1993. And the codes saying which kind of account to take the money from differ between published specifications that both describe themselves as ISO 8583: one major scheme's documentation uses `10` for savings and `20` for a current account, while a major acquirer's published message specification uses `01` and `02` for the same two things. The box number is agreed. The contents are negotiated.

The standard is silent about everything that makes it work. ISO 8583 says nothing about how a message gets from the terminal to the bank, how you know where one message ends and the next begins, how you prove it was not tampered with, or how the money actually moves afterwards. All of that is bolted on outside: routing headers borrowed from an older transport standard, a length prefix in front of every message on a TCP connection, a cryptographic checksum in the last field, and a separate settlement process hours or days later. Reading ISO 8583 and expecting to understand a card network is like reading the specification for a shipping container and expecting to understand global trade. It is also why `00` does not mean the money moved: it means the issuer agreed that it *would*, and the instruction that shifts funds between banks is a different message sent later, often the following day.

The technical version

What ISO 8583 is, precisely

ISO 8583 is an International Standard maintained by ISO Technical Committee 68 (*Financial services*), Subcommittee 9 (*Information exchange for financial services*). Its title is *Financial-transaction-card-originated messages — Interchange message specifications*. Its scope is one sentence: it "specifies a common interface by which financial-transaction-card-originated messages can be interchanged between acquirers and card issuers", and it "specifies message structure and format, including normalized data types". It then says explicitly that "the method by which messages are transported or settlement takes place is not within the scope of this document".

The editions are:

Edition	Designation	Published	Status as of writing
1	ISO 8583:1987	1987	Withdrawn
2	ISO 8583:1993	December 1993	Withdrawn 17 June 2003
—	ISO 8583-2:1998	1998	Withdrawn
—	ISO 8583-1:2003	June 2003	Withdrawn 18 July 2023
—	ISO 8583-3:2003	2003	Withdrawn
3	ISO 8583:2023	July 2023	Current

The third edition cancels and replaces all three parts published between 1998 and 2003 and reunifies them into a single document, and it does so by making that document much smaller: the messages, data elements and code values that used to occupy the bulk of ISO 8583-1:2003 are now published separately at `standards.iso.org/iso/8583` and maintained by an ISO 8583 maintenance agency, which as of writing is the Association française de normalisation (AFNOR). The foreword records the reason: the document "has been restructured to facilitate maintenance of the messages, data elements and code values by a new ISO 8583 maintenance agency".

The 2023 edition also carries a note that tells you what four decades did to the subject matter. It observes that "a range of tokens now exist for identifying account relationships", and warns that "the actual token numeric issued by a financial institution can be different from the associated card numeric". A standard about cards has quietly conceded that the thing in the message is frequently not a card. Its only normative references are ISO/IEC 7812-1 and ISO/IEC 7816-6.

Message structure

Clause 4.1.1 of the 2023 edition specifies that every message shall be constructed in exactly this sequence:

1. the message type;
2. one or two message bitmaps;
3. a series of data elements, in the order of the bitmap representation.

There is no framing, no terminator and no length. Those are the transport's problem, and every deployment solves them differently.

The message type identifier, digit by digit

The message type is four digits. The first is a version number; the remaining three are the message type identifier proper, which the standard describes as "a three-digit numeric field identifying the message class, message function and transaction originator".

Position 1 — version. The standard's Table 1 assigns:

Digit	Standard	Year
0	ISO 8583	1987
1	ISO 8583	1993
2	ISO 8583-1	2003
2	ISO 8583	2023
3 to 7	Reserved for ISO use	
8	Reserved for national use	
9	Reserved for private use	

The repetition of 2 is not an error, and it is the most quietly devastating line in the standard. The 2023 footnote explains that "code no. 2 is reused for this edition since no changes have been incorporated into this document that would necessitate a new version to properly construct or parse a message for processing". A new version digit is only assigned, per clause 4.1.2.2, "when sufficient changes have been made in a revision of this document such that it is necessary to know which version was used to construct a message in order to properly process the message". Thirty-six years on, ISO published a new edition and did not need a new digit: a standards body formally acknowledging that the wire format is frozen.

Position 2 — message class. What kind of business this message is about.

Digit	Class
1	Authorisation
2	Financial
3	File actions
4	Reversal and chargeback
5	Reconciliation
6	Administrative
7	Fee collection
8	Network management

The distinction between class 1 and class 2 is the dual-message versus single-message divide, and it decides the entire downstream architecture. An authorisation message asks whether funds are available and obtains an approval but posts nothing to the account; a separate clearing exchange posts it later. A financial message asks and posts in one go. Credit cards overwhelmingly run dual-message; ATM withdrawals and much domestic debit run single-message.

Position 3 — message function. What this message does within its class.

Digit	Function
0	Request
1	Request response
2	Advice
3	Advice response
4	Notification
5	Notification acknowledgement
6	Instruction
7	Instruction acknowledgement

The standard's definitions are worth quoting because the industry uses these words loosely. A *request* is a "message in which the sender informs the receiver that a transaction is in progress", and a response is required to complete the activity. An *advice* is a "message in which the sender notifies the receiver of an action that has been taken, requiring no approval but requiring a response". The difference is not politeness: a request can be refused, an advice can only be acknowledged. Functions 6 and 7 arrived in the 2003 edition and are almost never seen on live scheme links.

Position 4 — transaction originator.

Digit	Origin
0	Acquirer
1	Acquirer repeat
2	Issuer
3	Issuer repeat
4	Other
5	Other repeat

The repeat values matter more than they look. Clause 4.2 requires that "whenever a repeat message is identified, that repeat message shall be identical to its original message, with the exception of the message type identifier and, if necessary, date and time transmission and the message authentication code data elements". A repeat is a byte-for-byte retransmission with a changed origin digit, and that is how the receiver knows it may be a duplicate rather than a second purchase.

Reading them together. The four digits compose:

MTI	Reading
0100	1987, authorisation, request, from acquirer
0110	1987, authorisation, request response, from issuer
0120	1987, authorisation, advice, from acquirer
0130	1987, authorisation, advice response, from issuer
0200	1987, financial, request, from acquirer
0210	1987, financial, request response, from issuer
0220	1987, financial, advice, from acquirer
0230	1987, financial, advice response, from issuer
0400	1987, reversal, request, from acquirer
0410	1987, reversal, request response, from issuer
0420	1987, reversal, advice, from acquirer
0430	1987, reversal, advice response, from issuer
0800	1987, network management, request, from acquirer
0810	1987, network management, request response, from issuer
1240	1993, financial, notification, from other
1442	1993, chargeback, notification, from issuer

Note the last two. `1240` is Mastercard's First Presentment message and `1442` is a chargeback: both are 1993-version messages, and both travel over the same scheme as the 1987-version `0100` that authorised the purchase in the first place. One transaction, two editions of the standard, routinely.

The bitmap

The second component is "one or two message bitmaps, each consisting of 64 bits. Each bit signifies the presence (1) or the absence (0) in the message of the data element associated with that particular bit."

The primary bitmap is always present and indexes data elements 1 to 64. Bit 1 does not index a data element; it signals the presence of a secondary bitmap covering elements 65 to 128. In the 1993 and later editions, data element 65 can in turn signal a tertiary bitmap covering 129 to 192, which is exceptionally rare in production.

Take the message from the worked example, which carries data elements 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 37, 41, 42, 43 and 49, and no secondary bitmap. Writing a 1 in each of those positions and a 0 everywhere else:

```
0111 0010 0011 1100 0100 0100 1000 0001
0000 1000 1110 0000 1000 0000 0000 0000
```

Grouped into nibbles and written in hexadecimal, that is:

```
723C448108E08000
```

Eight bytes. Seventeen fields declared. No field names, no delimiters, no terminators, no ambiguity about order. Going the other way, `7010001102C04804` expands to

```
0111 0000 0001 0000 0000 0000 0001 0001
0000 0010 1100 0000 0100 1000 0000 0100
```

which declares elements 2, 3, 4, 12, 28, 32, 39, 41, 42, 50, 53 and 62 — a response message, carrying a card number, processing code, amount, local time, fee, acquirer identifier, response code, terminal, merchant, settlement currency, security control information and one private field. You can read the *shape* of a transaction off eight bytes before parsing a single value, which is why network engineers debug card traffic from hex dumps.

Data element notation

Each data element is defined by a content type and a length rule.

Notation	Content
a	Alphabetic, including spaces
n	Numeric only
s	Special characters only
an, as, ns, ans	The obvious combinations
b	Binary
z	Track 2 and track 3 code set, per ISO/IEC 7813 and ISO/IEC 4909
x+n	A sign character, `C` for credit or `D` for debit, followed by digits
p	Pad character

Length is either fixed or variable. A fixed field is written with its length: `n 6` is exactly six digits. A variable field carries its own length in a prefix, written as leading dots: `n..19` is an LLVAR field, a two-digit length followed by up to nineteen digits; `ans...999` is an LLLVAR field, a three-digit length followed by up to 999 characters. Clause 4.4.2 onwards distinguishes *primitive* elements, which have no internal structure; *constructed* elements, holding a fixed number of sub-elements all of which must be present; and, from 2003, *composite* elements, which contain datasets indexed by a second-level dataset bitmap or encoded as tag-length-value under ISO/IEC 8825.

The elements that carry a purchase

DE	Format	Name
2	n..19	Primary account number
3	n 6	Processing code
4	n 12	Amount, transaction

DE	Format	Name
7	n 10	Transmission date and time
11	n 6	System trace audit number
12	n 6	Local transaction time
13	n 4	Local transaction date
14	n 4	Expiration date
18	n 4	Merchant type
22	n 3	Point of service entry mode
25	n 2	Point of service condition code
32	n..11	Acquiring institution identification code
35	z..37	Track 2 data
37	an 12	Retrieval reference number
38	an 6	Authorisation identification response
39	an 2	Response code
41	ans 8	Card acceptor terminal identification
42	ans 15	Card acceptor identification code
43	ans 40	Card acceptor name and location
49	a or n 3	Currency code, transaction
52	b 64	Personal identification number data
55	ans...999	Integrated circuit card data
64	b 64	Message authentication code
90	n 42	Original data elements

Several of these repay a closer look.

DE3, processing code, is six digits in three pairs: transaction type, the account the money comes from, the account it goes to. `000000` is a purchase from the default account, `010000` a cash withdrawal, `200000` a refund, `300000` a balance enquiry. The account-type pairs are where implementations diverge most visibly, and the divergence is public: Mastercard's Single Message System material uses `00` for no account specified, `10` for savings, `20` for checking and `30` for a credit card account, while Worldpay's published Lync ISO 8583 message specification builds cash withdrawal from savings as `010100` and cash advance from credit as `010300`. Both describe ISO 8583. Neither is wrong. They are not the same dialect.

DE4, amount, is twelve digits with no sign and no decimal separator. The implied decimal position comes from the currency in DE49, an ISO 4217 code: `826` for pounds sterling and `840` for US dollars, both with two minor digits; `392` for Japanese yen, with none; `414` for Kuwaiti dinar, with three. A system that stores amounts as decimal strings and infers the exponent from the currency will survive; one that stores them as floating-point will not.

DE7, DE12 and DE13 are three different clocks. DE7 is the transmission date and time in MMDDhhmmss, expressed in Coordinated Universal Time by the sender of that particular message, and it changes at every hop. DE12 is the local transaction time at the point of sale, hhmmss, and DE13 the local transaction date, MMDD. Neither carries a year in the 1987 edition, so a system reconstructing a full timestamp has to supply the year from somewhere else, and the last week of December is where that assumption goes to die.

DE11, the system trace audit number, is six digits assigned by the message originator. It is not globally unique and was never meant to be: it is unique within one acquirer, within one processing day, and it wraps at 999999. Its job is to let the two ends of one link match a response to a request.

DE37, the retrieval reference number, is twelve alphanumeric characters and is the identifier that survives longest into the dispute process. Its internal construction is not fixed by ISO; a common convention is a single-digit year, a three-digit day of year, a two-digit hour and a six-digit sequence, which is how the example `607614430041` was built: year 6, day 076 (17 March), hour 14, sequence 430041.

DE38, the authorisation identification response, is the six-character approval code the issuer generates and the receipt prints. **DE39** is the response code: two alphanumeric characters in the 1987 edition, three in 1993.

DE55 is where EMV lives, and it is a format inside a format. The 2023 standard is explicit that the composite dataset structure "does not apply to the integrated circuit card (ICC)-related data element, as the linking of related sub-elements shall be accomplished in accordance with the definitions given in ISO/IEC 7816-6". DE55 is therefore a bag of EMV tag-length-value objects — the application cryptogram in tag 9F26, the cryptogram information data in 9F27, the application transaction counter in 9F36, the terminal verification results in 95, the unpredictable number in 9F37 — parsed by entirely different rules from the message carrying them, and commonly capped by schemes far below the ISO maximum of 999 bytes.

DE52 carries the PIN, always enciphered, as a 64-bit block formatted according to ISO 9564-1. Format 0 blocks, which combine the PIN with digits of the account number, dominate the installed base; format 4 exists to allow AES. **DE64** and its secondary-bitmap twin DE128 carry a message authentication code computed with a shared key, so the receiver can detect tampering in transit.

DE90, original data elements, is forty-two digits and is a constructed field in the strict sense: five sub-elements, all mandatory, no length prefixes, laid out end to end.

Sub-element	Digits	Content
Original MTI	4	The message being reversed
Original DE11	6	Its system trace audit number
Original DE7	10	Its transmission date and time
Original DE32	11	Its acquiring institution code
Original DE33	11	Its forwarding institution code

Four plus six plus ten plus eleven plus eleven is forty-two. DE90 is how a reversal names its victim, and the fact that it needs five separate fields to do so tells you something true and slightly alarming about how transactions are identified in this format.

Why a bitmap rather than a text format

The engineering argument has four parts, and only the first is about size.

Size. The worked authorisation above, encoded with ASCII digits and a hexadecimal bitmap, is 181 bytes; with packed binary-coded decimal numerics and a raw binary bitmap it is 129. The same content as tagged text with descriptive names is 639 bytes, and with the tags reduced to bare field numbers it is 289. On a 1,200 bit-per-second dial-up link with start and stop bits — ten bits per character — that is the difference between roughly eleven seconds of round trip and roughly two. On a 9,600 bit-per-second X.25 leased line shared between hundreds of terminals, it is the difference between a switch that copes at the Saturday peak and one that does not.

Determinism. A bitmap-indexed message can be parsed with a fixed loop, a lookup table and no dynamic memory allocation whatsoever. There is no tokeniser, no nesting, no escaping, no quoting

and no ambiguity about whether a value is a string or a number, because the field definition already said. A terminal with sixteen kilobytes of RAM could parse this in 1987, and a switch handling tens of thousands of messages a second can parse it now without a garbage collector ever running.

Bounded worst case. Every field has a maximum length declared in the specification, so the largest possible message is computable in advance and buffers can be statically sized. Text formats have no such property.

Honesty about what it costs. All four of those advantages come from positional encoding, and positional encoding has one enormous liability: the message is not self-describing. A malformed bitmap does not produce a parse error at the malformed byte; it produces a message in which every subsequent field is read at the wrong offset with the wrong length, and the failure surfaces somewhere else entirely, often as a nonsense amount. Anyone who has debugged card traffic has seen a £42.50 purchase arrive as £4,250,000 because one field upstream was written one digit short.

The three versions, and why 1987 will not die

1987, the first edition, defined 128 data elements, a two-character response code in DE39, and DE24 as a network international identifier used for routing.

1993 kept the message structure identical and changed the vocabulary. The most consequential change was to response codes, which became three digits arranged so that the leading digit tells you which class of message the code belongs to:

Range	Meaning
000–099	Approved
100–199	Denied, card need not be retained
200–299	Denied, card to be retained
300–399	File action result
400–499	Reversal or chargeback result
500–599	Reconciliation result
600–699	Administrative result
700–799	Fee collection result
800–899	Network management result
900–999	Could not be processed

So `000` is approved, `100` is do not honour, `101` is expired card, `116` is insufficient funds, `200` is do not honour and retain the card, `911` is issuer timed out. Within each hundred, the standard reserves blocks for ISO, national and private use, so a scheme can invent codes without colliding with the standard. It is a better design than the 1987 flat two-digit list in every respect. DE24 was redefined as a function code, and the data element count rose to a possible 192 through a tertiary bitmap.

2003 went further and broke things. Currency ceased to be a standalone data element and became a sub-element of each amount, which is arguably correct and which invalidates every parser written for the previous sixteen years. Composite data elements and datasets arrived, along with the option of encoding sub-elements as tag-length-value under ISO/IEC 8825.

2023 restructured the document, moved the definitions to a maintenance agency, folded in accumulated corrections approved by the registration and maintenance management group, and reused version digit 2.

Deployment tells a different story from publication. Visa's BASE I technical specifications, which define the authorisation messages of the V.I.P. system, are built on the 1987 edition with a large body of Visa-specific additions. Mastercard's Customer Interface Specification carries authorisations as 1987-version messages; its Integrated Product Messages clearing formats carry presentments and chargebacks as 1993-version messages such as the First Presentment/1240. The 2003 edition, twenty-three years after publication, never achieved meaningful adoption on scheme links at all.

The reason is not inertia in the pejorative sense. It is arithmetic. A card network is a mesh of bilaterally certified endpoints — acquirers, processors, gateways, issuer hosts, ATM estates, terminal fleets — numbering in the tens of thousands, and every pair that exchanges messages has been through a certification process that produced a signed test result. A change to the wire format requires every one of those pairs to re-certify, and a flag day on which both ends switch simultaneously, because a 1987 parser handed a 2003 message does not degrade gracefully; it misreads the amount. The value of the newer edition is real but marginal, and the cost of the transition is catastrophic. So the industry stayed on 1987 and expressed everything new in private-use fields, which brings us to the part that actually costs money.

What "ISO 8583 compliant" is worth

Roughly nothing, on its own. The right question is never "is it ISO 8583?" but "which specification, which version of that specification, and have you got the certification letter?"

Every network of consequence publishes its own document. A partial list of the names, which is useful because these documents are what integration projects actually work from:

Network	Specification
Visa	BASE I Technical Specifications; Single Message System ATM/POS Technical Specifications; Dual Message System Authorization Technical Specifications
Mastercard	Customer Interface Specification; Global Clearing Management System Reference; Integrated Product Messages Clearing Formats; Mastercard Debit Switch Online Specifications; Single Message System Specifications
American Express	Global Credit Authorization Guide; Global Electronic Data Capture Terminal Message Specification; Plural Interface Processing Terminal Interface
Discover	Authorization Interface Technical Specifications
JCB	J/Link Online Interface Guide
UnionPay	CUP Online Message
Diners Club	DCISC Relay Authorisation System; Relay Xpress Authorization System

They diverge in ways that go well past field contents. Visa's BASE I messages are normally carried with a 22-byte header whose values determine the format variant of certain data elements, and BASE I places data elements in three bitmapped arrays of 64 elements each, with all three bitmaps positioned together after the message type and before the message body — rather than deriving the second and third from bit 1 of the primary and bit 1 of DE65, as the standard describes. A parser written strictly to ISO will not read a BASE I message at all.

Mastercard carries the bulk of its proprietary content in private data. In authorisation messages that content lives in numbered subelements of DE48; in clearing messages it lives in private data subelements, the four-digit PDS tags such as PDS 0032 for enhanced acceptor name or PDS 0198 for device type indicator. The same business fact appears in a different container depending on whether you are looking at the authorisation or the clearing record, which is a large part of why reconciliation between the two is the work it is.

Below the schemes, national and vendor dialects abound. Australia's AS 2805 series incorporates ISO 8583 and extends it across a far wider range of payment topics; processor platforms such as ACI's BASE24 define their own external message formats, ISO 8583 in shape and proprietary in content. Data elements 60 to 63 and 120 to 127 are reserved for private use in the standard itself, which is its way of saying: put your incompatibilities here, and stop asking us.

The practical consequence is that conformance is not a property of your code. It is a property of a certification event between your endpoint and one named counterparty, on one named specification, at one named version, evidenced by a test script both sides signed. "We are ISO 8583 compliant" is a statement about a shape.

Message flow

Request and response. The acquirer sends `0100`; the issuer answers `0110`. The response echoes the fields the two sides use to match it to the request — conventionally DE7, DE11, DE32 and DE37, though the precise matching key is set by each scheme and not by ISO — and adds DE38 and DE39. The acquirer runs a timer, and if it expires the acquirer does not know whether the issuer approved, because a response can be lost after the issuer has already committed a hold. That ambiguity is the origin of most of what follows.

Repeat. On a timeout, the acquirer may retransmit with origin digit 1 instead of 0, producing `0101`. Per clause 4.2 the repeat must be byte-identical apart from the MTI, the transmission date and time and the MAC. The issuer, seeing the same STAN and retrieval reference number, is expected to return the original decision rather than authorise a second time.

Reversal. If the acquirer concludes that the transaction did not complete — timeout after retries, a customer who removed the card, a till that crashed mid-sale — it sends `0400`, quoting DE90 so the issuer can find the original, and expects `0410`. Reversals are retried persistently, sometimes for days, because an unreversed authorisation leaves the cardholder's available balance reduced against a purchase that never happened. Where the acquirer is stating a fact rather than asking permission, it uses the advice form `0420`, acknowledged with `0430`.

Advice. Advices exist for everything that happened before anyone could ask. A terminal that authorised offline because the amount fell below its floor limit sends `0220`. A transaction approved by the scheme on the issuer's behalf while its host was unreachable — stand-in processing — reaches the issuer as an advice, because by then the answer has already been given to the merchant.

Completion. A pre-authorisation taken at a hotel check-in or a fuel pump is `0100` for an estimated amount; the final amount arrives later as a financial advice `0220`, which is why a hotel hold can sit on a card for days after checkout.

Network management. `0800` and `0810` keep the link itself alive. DE70, the network management information code, distinguishes the purposes; the values almost universally implemented are `001` for sign-on, `002` for sign-off and `301` for an echo test. Echo tests run every few seconds on a live link, and a switch that stops answering them is taken out of rotation before any customer notices. When you

read that a card network was "down for eleven minutes", the detection almost certainly happened in an `0800`.

Reconciliation. Class 5 messages carry the totals each side believes it exchanged during a settlement window — counts and amounts of credits, debits, reversals and fees, in data elements 74 to 89.

One authorisation, framed for the wire

What actually leaves a terminal is not an ISO 8583 message. It is an ISO 8583 message wrapped in whatever the link demands. A typical arrangement over TCP:

Layer	Bytes	Content
Length prefix	2	Binary count of the bytes that follow
Transport header	5	Protocol identifier `60`, destination address, source address
Message type	4	`0100`
Primary bitmap	8 or 16	`723C448108E08000`
Data elements	remainder	In bitmap order

The five-byte transport header is a Transaction Protocol Data Unit, inherited from an older transport standard and kept because ISO 8583 itself contains no routing information at all. Its convention is one byte of protocol identifier, conventionally `60`, then two bytes of destination address and two of source address; the response is formed by swapping the two addresses. None of it is in ISO 8583. All of it is mandatory in practice.

Where it runs out

The pressure on ISO 8583 is not performance. Switches process these messages in tens of microseconds. It is that the world now wants to say things the format has no room for: full merchant and payee details for sanctions screening, structured remittance information, extended tokenisation data, richer fraud signals, and the growing bulk of EMV data in DE55. The industry's answer for thirty years has been to overload private fields with sub-formats — subelements inside DE48, PDS tags inside clearing records, TLV inside DE55 — until a message contains three different encoding schemes nested inside each other and the integration cost of any change is measured in quarters.

That is the case for ISO 20022, and the subject of a later chapter. ISO 8583 is not being replaced because it failed, but because it succeeded so completely that a format designed to fit a purchase down a 1,200 bit-per-second telephone line is now the constraint on what a purchase is allowed to mean.

The next chapter takes the bitmap on its own and shows, bit by bit, what those sixty-four flags do to the parser at the other end.

24.98 Common wrong ideas

Wrong: ISO 8583 is the language of card payments. **Right:** it is a filing convention specifying the shape of the envelope, and since 2023 the standard does not even contain the message, field and code definitions, which now live in a separately maintained register.

Wrong: two systems that are both faithful to ISO 8583 can exchange messages. **Right:** faithfulness settles nothing about what goes in DE 48 or DE 62, or what response code `05` means to you as opposed

to me; conformance is a property of a certification event between two named endpoints, not of your code.

Wrong: ISO 8583 is a binary format. **Right:** only the bitmap is genuinely binary; the rest is usually characters in ASCII or EBCDIC, and because the standard permits both packed and unpacked numerics the same logical message is 129 bytes on one link and 181 on another.

Wrong: a data element number means the same thing in every edition. **Right:** DE 24 is a network routing identifier in 1987 and a function code in 1993, and response codes are two characters in 1987 and three in 1993.

Wrong: a response code of `00` means the money moved. **Right:** it means the issuer agreed that it would; the instruction that shifts funds between banks is a different message sent later, often the following day.

Wrong: the amount field carries its own currency and decimal point. **Right:** DE 4 is twelve unsigned digits with no separator, and the implied decimal position comes from the ISO 4217 code in DE 49 — two places for sterling, none for yen, three for the Kuwaiti dinar.

Wrong: a repeat message is simply another attempt. **Right:** clause 4.2 requires it to be identical to its original apart from the message type identifier, the transmission date and time and the MAC, which is precisely how the receiver knows it may be a duplicate rather than a second purchase.

Wrong: DE 11 uniquely identifies a transaction. **Right:** the system trace audit number is unique only within one acquirer within one processing day and wraps at 999999; DE 37 is the identifier that survives into the dispute process.

Wrong: a parser written strictly to the standard will read any scheme's traffic. **Right:** Visa's BASE I carries a 22-byte header and places three bitmapped arrays together after the message type rather than deriving them from bit 1, so a strictly conformant parser will not read a BASE I message at all.

Wrong: the industry stays on the 1987 edition out of inertia. **Right:** it is arithmetic — tens of thousands of bilaterally certified endpoint pairs would each have to re-certify and switch on a flag day, because a 1987 parser handed a 2003 message does not fail cleanly, it misreads the amount.

24.99 Chapter summary in 20 lines

1. ISO 8583 is not a language but a filing convention: an agreement about which numbered box holds which value, and how to tell the other end which boxes you bothered to fill in.
2. The whole trick is the bitmap, sixty-four bits declaring the presence or absence of each data element, so that no field name ever has to travel.
3. That compression was necessity rather than elegance: at 1,200 bits per second the difference between labelled text and numbered boxes was the difference between a customer waiting eleven seconds and waiting two.
4. Every message is a message type, one or two bitmaps, and then the data elements in bitmap order, with no framing, no terminator and no length, all of which are the transport's problem.
5. The four-digit message type indicator encodes version, class, function and origin, and reading it digit by digit tells you what kind of message you are holding before you parse anything else.
6. There are really only four kinds of message: a question that may be refused, an answer, an advice that may only be acknowledged, and a cancellation that quotes the message it wants undone.

7. The 2023 third edition moved the messages, data elements and code values out of the standard into a register held by a maintenance agency, so that they can change without republishing an International Standard.
8. It also reused version digit 2, formally conceding that nothing had changed in a way that affects how a message is constructed or parsed.
9. The standard is silent about transport, framing, integrity and settlement, so every deployment bolts on a length prefix, a routing header inherited from an older standard, and a cryptographic checksum.
10. Data elements are defined by a content type and a length rule, fixed or variable, with the variable ones carrying two- or three-digit length prefixes of their own.
11. DE 4 carries twelve digits with no decimal separator, and the exponent comes from the ISO 4217 currency code in DE 49, which is why floating-point storage of amounts does not survive contact with this format.
12. DE 7, DE 12 and DE 13 are three different clocks — universal time rewritten at every hop, and a local time and date carrying no year at all.
13. DE 11 matches a response to a request across one link, while DE 37 is the twelve-character handle that survives into clearing and into a dispute months later.
14. DE 55 is a format inside a format: EMV tag-length-value objects linked according to ISO/IEC 7816-6, carried inside a message parsed by entirely different rules.
15. DE 90 names the victim of a reversal with five mandatory sub-elements totalling forty-two digits, which says something true and slightly alarming about how transactions are identified here.
16. The bitmap buys size, deterministic parsing, a bounded worst case and no dynamic allocation, at the single cost that the message is not self-describing.
17. A malformed bitmap therefore does not fail at the malformed byte; every later field is read at the wrong offset, and a £42.50 purchase arrives as £4,250,000.
18. The 1993 edition's three-digit response codes, classed by leading digit, are better than the 1987 flat list in every respect and still lost, because re-certifying the mesh costs more than the improvement is worth.
19. Every network of consequence publishes its own specification, and the divergences run past field contents into bitmap layout and private data, so "we are ISO 8583 compliant" is a statement about a shape.
20. The format is not being replaced because it failed but because it succeeded so completely that a design fitted to a 1,200 bit-per-second telephone line is now the constraint on what a purchase is allowed to mean.

Chapter sources: ISO 8583:2023, third edition, clauses 1, 3, 4 and 7 and Table 1; ISO catalogue entries for ISO 8583:1993, ISO 8583-1:2003 and ISO 8583:2023; the ISO register of maintenance agencies; ISO 4217; ISO 9564-1; ISO/IEC 7812-1 and ISO/IEC 7816-6; Mastercard Switch Rules, Transaction Processing Rules, Single Message System documentation and Mastercard Developers material for message types 1240 and 1442, DE48 subelements and PDS tags 0032 and 0198; Worldpay Lync ISO 8583 Message Specification v1.3 for processing codes; iso8583.info for the Visa BASE I message structure and the scheme specification names; EFTlab for the Transaction Protocol Data Unit; and the Wikipedia ISO 8583 article for the consolidated 1987 and 1993 data element and response code tables, cross-checked against the standard text where possible.

3

CHAPTER

The Bitmap

25.0 What this chapter gives you

1. You will be able to read a sixteen-character hexadecimal string such as `F23C448128E09200` and say, without a tool, which data elements the message carries.
2. You will be able to explain why bit 1 is not a field at all, and why getting it wrong shifts every subsequent byte in the message by eight or sixteen.
3. You will be able to tell the three wire forms of the same sixty-four bits apart — raw binary, ASCII hexadecimal and EBCDIC hexadecimal — by inspecting the first sixteen bytes of a link.
4. You will be able to write the receive path for an ISO 8583 message in about twenty lines, and say why it needs no tokeniser, no state machine and no backtracking.
5. You will be able to read the notation `n 6`, `n..19` and `ans...999` for what it says about length, and read `b 64` as eight bytes rather than sixty-four.
6. You will be able to diagnose a misaligned parse from its symptom — a card number of all zeroes, an amount with two extra digits, a MAC failure — because there will be no error message to work from.
7. You will be able to say why sixty-four bits is exactly optimal in the worst case, gives up about a quarter against an ideal coder on real traffic, and beats a list of field numbers from about the tenth field onward.
8. You will be able to explain why an unknown field is survivable in Protocol Buffers and fatal in ISO 8583, and why that one property makes adding a field to a card network a certification programme rather than a deployment.
9. You will be able to use a bitmap as an instrument: classify traffic before decoding any field, and name the exact difference between a request and its response with a single exclusive-or.
10. You will be able to state where the bitmap runs out — 128 slots, no versioning, no room for meaning — and why the industry answered with nested structures inside elements 48, 62, 63, 124 and 127 rather than with the tertiary bitmap.

There is a small piece of engineering at the front of every card message on earth that almost nobody outside the payments industry has heard of, and that almost nobody inside it can explain properly. It is eight bytes long. It contains no card number, no amount, no merchant, no date, and no information about the transaction at all. What it carries is information about the *message* — specifically, which of the message's sixty-four possible fields the sender bothered to fill in.

It is called the bitmap. It was designed in the mid-1980s for terminals with kilobytes of memory talking over telephone lines at 1,200 bits per second. Those constraints are gone. The bitmap is not. It is still there, at the front of authorisations approved in 2026, doing the same job in the same eight bytes, and

the reason is not inertia. The reason is that for the problem it solves, sixty-four bits is very close to the smallest number of bits that could possibly work.

This chapter takes those eight bytes apart. By the end of it you will be able to look at a hexadecimal string in a log file and say, without a tool, which fields are in the message.

The plain version

Imagine a class of sixty-four children who always sit in the same order — the same sixty-four chairs, the same sixty-four names, every day, for forty years. Nobody ever moves.

Every morning the class hands in homework, and there is a rule: nobody writes their name on it. The teacher collects the pile, the pile is in seat order, and that is the only thing identifying whose is whose. Most days most of the class has not done any, so on a typical morning maybe twenty children hand something in and the pile is twenty sheets deep, in seat order, with no names.

How does the teacher know the third sheet belongs to child eleven and not child nine?

Because at the top of the pile there is a register. Not a list of names — a row of sixty-four boxes, one per seat, in seat order. A tick means "this child handed something in". A blank means "skip this seat". The teacher reads the register first, learns that seats 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 35, 37, 41, 42, 43, 49, 52 and 55 have ticks, and deals the pile out in that order. Twenty sheets, twenty owners, no names written anywhere.

That row of ticks is the bitmap. Everything in this chapter follows from it.

Ticks are expensive; symbols are cheap

Writing out sixty-four boxes takes a lot of space, so we compress it with a trick that takes five minutes to learn. Take the ticks four at a time. Four boxes can be filled in sixteen different ways: all blank, only the last one, only the third one, and so on up to all four. Invent sixteen symbols, one per pattern, and write one symbol instead of four boxes. The symbols are `0` to `9` and then `A`, `B`, `C`, `D`, `E`, `F`. This is called hexadecimal, and the whole of it fits in a small table.

Symbol	Ticks	Symbol	Ticks
`0`	blank blank blank blank	`8`	tick blank blank blank
`1`	blank blank blank tick	`9`	tick blank blank tick
`2`	blank blank tick blank	`A`	tick blank tick blank
`3`	blank blank tick tick	`B`	tick blank tick tick
`4`	blank tick blank blank	`C`	tick tick blank blank
`5`	blank tick blank tick	`D`	tick tick blank tick
`6`	blank tick tick blank	`E`	tick tick tick blank
`7`	blank tick tick tick	`F`	tick tick tick tick

There is a pattern that makes this easy without the table. Left to right, the four boxes are worth 8, 4, 2 and 1; add up the ticked ones. Blank, tick, tick, blank is 4 plus 2, which is 6. Tick, tick, tick, blank is 8 plus 4 plus 2, which is 14, and 14 is `E`. That is the entire system.

Sixty-four boxes, four to a symbol, is sixteen symbols. So the register — the thing that tells you the shape of an entire card payment — is sixteen characters long. That is shorter than most people's email address.

One real register, decoded by hand

Here is a real one, from a chip-and-PIN purchase:

```
F23C448128E09200
```

Decode it left to right, four boxes per symbol, keeping a running count of which seat we are on.

Symbol	Ticks	Seats covered	Ticked
F	1111	1 to 4	1, 2, 3, 4
2	0010	5 to 8	7
3	0011	9 to 12	11, 12
C	1100	13 to 16	13, 14
4	0100	17 to 20	18
4	0100	21 to 24	22
8	1000	25 to 28	25
1	0001	29 to 32	32
2	0010	33 to 36	35
8	1000	37 to 40	37
E	1110	41 to 44	41, 42, 43
0	0000	45 to 48	none
9	1001	49 to 52	49, 52
2	0010	53 to 56	55
0	0000	57 to 60	none
0	0000	61 to 64	none

Twenty-one seats are ticked: 1, 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 35, 37, 41, 42, 43, 49, 52 and 55.

Now put the names back. Seat 2 is the card number, 3 is what kind of transaction, 4 is the amount, 7 is when it was sent, 11 is the serial number. Seats 12 and 13 are the shop's own clock and calendar; 14 is the card's expiry; 18 is what kind of shop; 22 is how the card was read; 25 is anything unusual; 32 is which bank is asking; 35 is the data copied off the card; 37 is a reference number. Seats 41, 42 and 43 are the till, the merchant and the merchant's name and town. Seat 49 is the currency, 52 is the encrypted PIN, and 55 is everything the chip said.

You have just read a card payment out of sixteen characters. You know it was a chip transaction with a PIN, in a shop, from a real till, with a real merchant name attached. You have not seen a single value yet.

Seat one is not a child

Look at that list again. Seat 1 is ticked, but nothing in the description above is seat 1, because seat 1 is not a child. It is a note that says *there is a second register*. Sixty-four seats was enough in 1985 and it was not enough by 1990, so the designers reserved the very first tick to mean "another sixty-four boxes follow, covering seats 65 to 128". If seat 1 is blank, the register is sixteen characters and that is all. If seat 1 is ticked, the register is thirty-two characters, and the second half describes a second class of sixty-four children who are only occasionally in the room.

In our example the second register is:

```
0000000000000001
```

Fifteen zeroes and a one. Read the same way, the final `1` is the last box of the last group of four, which is seat 128 — the cryptographic seal proving the message was not altered on the way. So the full register for this payment is thirty-two characters:

```
F23C448128E09200 0000000000000001
```

and it declares twenty-two things: bit 1 saying "look at the second half", the twenty fields of the transaction, and the seal at the end.

Some homework is one page and some is forty

There is one more problem the register does not solve. It tells you *whose* homework is in the pile. It does not tell you *how many pages* each piece is.

For most seats it does not matter, because the answer is fixed and everyone knows it. Seat 4, the amount, is always exactly twelve digits. Seat 11, the serial number, is always exactly six. Seat 37 is always exactly twelve characters. You never have to ask. You count.

But some fields cannot be fixed. A card number can be thirteen digits or nineteen. A merchant's extra notes might be four characters or four hundred. For those, the field carries its own page count on the front. A card number arrives as the two digits `16` followed by sixteen digits. Chip data arrives as the three digits `120` followed by 120 bytes. Two digits of length at the front is called LLVAR, three digits is LLLVAR, and that is the whole naming convention: one letter `L` per digit of length, then the word "variable".

So the reader's job is a loop with no cleverness in it at all. Read the register. For each ticked seat, in order: if it is fixed, take exactly that many characters; if it is variable, read two or three digits of length first, then take that many. Move on. Stop when the ticks run out.

What happens when the register is wrong

Now the thing that makes this design frightening. Suppose the register ticks seat 25, but the child in seat 25 did not hand anything in. The teacher takes the next two characters and calls them seat 25's homework — but those two characters were the first two of seat 32's. Seat 32 is now short by two and reads two characters of seat 35. Seat 35 is variable length, so the teacher now reads a *length* out of the middle of somebody else's data, and that length might say 87 when 200 characters remain, or it might say 04.

Every field after the mistake is wrong. Not missing. Wrong. And nothing in the message says so, because there is no name on any of the homework to contradict the register: the whole point was that we stopped writing names to save space.

That is the bargain. The bitmap buys the smallest possible message and the fastest possible reader, and the price is that a single wrong bit does not produce an error where the error is. It produces a plausible-looking message full of nonsense, and the nonsense surfaces somewhere else entirely — often as an amount with two extra zeroes on it.

Where the plain version stops being true

The register is not a register of children, and the numbering is self-referential. In the standard's own numbering the bitmap *is* data element 1, so bit 1 of data element 1 indicates the presence of a second instance of data element 1. That is not a curiosity: it is why you will see published field tables listing element 1 as `b 16` and others listing it as `b 64`, describing the same object. One is counting hexadecimal characters, the other bits, and neither says which. The same ambiguity infects every binary element

in the catalogue. When a specification says `b 64` it almost always means sixty-four *bits*, which is eight bytes, which is sixteen hexadecimal characters. Reading it as sixty-four bytes is a classic first-week mistake.

Sixteen characters and eight bytes are both true, and they are not the same number of bytes on the wire. The plain version treats the bitmap as "sixteen characters". That is one of at least three encodings in real use. The bitmap can travel as eight raw binary bytes, whose individual bits carry the meaning directly, or as sixteen printable characters in ASCII, or as sixteen printable characters in EBCDIC on IBM mainframe links. All three carry identical information, occupy eight, sixteen and sixteen bytes respectively, and look nothing like each other on the link. A parser configured for the wrong one is not slightly wrong; it is catastrophically wrong from the first byte.

A tick tells you a field is present. It does not tell you where it is. The bitmap is not an index. There is no offset table, no pointer, no random access. To find data element 55 you must parse elements 2 through 43 first, because several of them are variable-length and you cannot know where 55 begins until you have measured everything before it. The bitmap makes the message small and the parse deterministic. It does not make the message seekable.

The bitmap says a field exists. It never says what the field means. Bit 62 set tells a parser that data element 62 is present and, from the local field table, how long it is. It says nothing about the contents, because 62 is reserved for private use and is defined by whoever you happen to be connected to. A parser holding the wrong counterparty's field table will read a syntactically valid message and produce semantically wrong output, silently. Worse: for a private element whose length rule the parser does not have, it cannot even skip the field, because it does not know how many bytes to skip. This is the deep limitation of positional encoding, and no care in the bitmap fixes it.

The technical version

What the standard specifies

ISO 8583 defines the message as a message type identifier, followed by one or more bitmaps, followed by the data elements that the bitmaps declare, in ascending order of element number. The third edition, published in July 2023, describes the bitmap component as "one or two message bitmaps, each consisting of 64 bits", in which "each bit signifies the presence (1) or the absence (0) in the message of the data element associated with that particular bit".

Three properties follow from that sentence and they are the whole design. The bitmap is *fixed width*: sixty-four bits always, whether the message carries three data elements or fifty, so a bitmap needs no length prefix of its own. It is *positional*: bit n corresponds to data element n, with no lookup, no translation table and no negotiation, the correspondence being a property of the numbering itself. And it is *ordering*: the data elements follow in ascending numerical order, so the bitmap answers "which fields" and "in what sequence" with the same sixty-four bits. That is why no separators are needed anywhere in the message.

Bit numbering, and the convention people get wrong

Bits are numbered from 1 to 64, left to right, starting at the most significant bit of the first byte. Bit 1 is `0x80` of byte 0, bit 8 is `0x01` of byte 0, bit 9 is `0x80` of byte 1. Formally, for element n the byte index is `(n - 1) / 8` and the mask is `0x80 >> ((n - 1) % 8)`.

This is big-endian bit ordering within the byte, and it is the most common source of a first implementation that almost works. A developer whose instinct is to test `byte & (1 << bit)` — least significant bit first, the convention used almost everywhere else in computing — will produce a parser that reverses

every group of eight, and the failure is not obvious, because the reversed set is still a plausible-looking set of field numbers. Byte `0xF2` correctly declares elements 1, 2, 3, 4 and 7; read least-significant-bit-first it declares 2, 5, 6, 7 and 8. Both look like field numbers. Only one is right.

One bitmap, decoded completely

Take the primary bitmap from earlier — a `0100` authorisation request for a contact chip transaction with online PIN — and expand it fully, in binary, with the element number each bit governs written alongside:

```
byte 0  F2  1111 0010   elements  1  2  3  4 |  5  6  7  8
byte 1  3C  0011 1100   elements  9 10 11 12 | 13 14 15 16
byte 2  44  0100 0100   elements 17 18 19 20 | 21 22 23 24
byte 3  81  1000 0001   elements 25 26 27 28 | 29 30 31 32
byte 4  28  0010 1000   elements 33 34 35 36 | 37 38 39 40
byte 5  E0  1110 0000   elements 41 42 43 44 | 45 46 47 48
byte 6  92  1001 0010   elements 49 50 51 52 | 53 54 55 56
byte 7  00  0000 0000   elements 57 58 59 60 | 61 62 63 64
```

Set bits: 1, 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 35, 37, 41, 42, 43, 49, 52, 55.

Bit 1 is set, so a secondary bitmap follows immediately, before data element 2.

```
00 00 00 00 00 00 00 01
```

The only set bit is the last bit of byte 7, which is bit 64 of that bitmap, which is data element 128.

On the wire, then: MTI, primary bitmap, secondary bitmap, and then elements 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 35, 37, 41, 42, 43, 49, 52, 55 and 128. Four are variable length — the primary account number in element 2 (`n..19`), the acquiring institution identification code in 32 (`n..11`), the track 2 data in 35 (`z..37`) and the integrated circuit card data in 55. The rest are fixed: `n 12` for the amount in 4, `n 6` for the system trace audit number in 11, `an 12` for the retrieval reference number in 37, `ans 8` for the terminal identification in 41, `b 64` for the PIN block in 52 and for the message authentication code in 128.

Element 55 is worth a warning. Published tables disagree: the widely circulated 1987 catalogue lists it as `ans ...999`, while scheme-facing references describe it as binary with a three-digit length, which is what it is in every EMV deployment — a concatenation of EMV tag-length-value objects, which are binary and are not character data. Take the counterparty's specification, not the general table.

Going the other way

To construct a bitmap, start with eight zero bytes and set one bit per element: element 2 sets `0x40` in byte 0, element 39 sets `0x02` in byte 4, element 64 sets `0x01` in byte 7. Two published bitmaps make good test vectors. `7010001102C04804` expands to elements 2, 3, 4, 12, 28, 32, 39, 41, 42, 50, 53 and 62 — a response carrying a card number, a processing code, an amount, a local time, a transaction fee, an acquirer identifier, a response code, a terminal, a merchant, a settlement currency, security-related control information and one private field. `7234054128C08000` expands to elements 2, 3, 4, 7, 11, 12, 14, 22, 24, 26, 32, 35, 37, 41, 42 and 49 — a magnetic stripe authorisation from an era when element 24 still carried a network international identifier. If your parser reproduces both, your bit arithmetic is right.

The secondary bitmap

The secondary bitmap covers data elements 65 to 128. It is present if and only if bit 1 of the primary bitmap is set, and when present it sits immediately after the primary bitmap, before any data element value.

The consequence is that the secondary bitmap is not optional in the way other fields are optional. Its presence or absence changes the byte offset of every subsequent field in the message by eight or sixteen bytes. Getting bit 1 wrong is the highest-blast-radius single-bit error in the whole format.

Elements 65 to 128 hold what a purchase does not need: settlement and reconciliation totals in 74 to 89, the original data elements of a reversal in 90, the file-action elements in 91 to 94, the settlement and receiving institution identifiers in 99 and 100, and private-use space in 120 to 127. Element 96 holds a message security code and 128 a message authentication code.

A `0400` reversal request typically sets element 90, "Original data elements", a forty-two digit fixed field holding the original message's MTI, system trace audit number, transmission date and time, and acquiring and forwarding institution identification codes, concatenated. That is how a reversal names the transaction it is reversing when there is no other shared identifier. A reversal carrying element 90 and a MAC in 128 has a secondary bitmap of:

```
00 00 00 40 00 00 00 01
```

Byte 3 is `0x40`, whose second bit is bit 26 of the secondary bitmap, which is element 90. Byte 7 is `0x01`, element 128.

The tertiary bitmap

Later editions extend the element range to 192, with a tertiary bitmap covering 129 to 192 signalled by element 65. The 1987 catalogue lists element 65 as `b 1`, an extended bitmap indicator — a single bit whose whole purpose is to say that a third bitmap follows.

You will not see one. Every reference that discusses it describes it as extremely rare or effectively unused on card rails, because the schemes solved the same problem a different way, by defining private elements that contain their own internal structure. Mastercard's private data subelements inside data elements 48, 124 and 127, and Visa's proprietary subfields inside 48, 62 and 63, are nested tag-length-value structures with their own subelement identifiers. They gave the industry unbounded extension space without ever touching the top-level bitmap.

Three wire forms of the same sixty-four bits

The bitmap can be represented as eight bytes of binary data or as sixteen hexadecimal characters in ASCII or EBCDIC. This is not an implementation detail. It is the first thing you must establish about any link, and it is not carried in the message. The same primary bitmap, in all three:

Form	Bytes on the wire	Length
Raw binary	`F2 3C 44 81 28 E0 92 00`	8
ASCII hexadecimal	`46 32 33 43 34 34 38 31 32 38 45 30 39 32 30 30`	16
EBCDIC hexadecimal	`C6 F2 F3 C3 F4 F4 F8 F1 F2 F8 C5 F0 F9 F2 F0 F0`	16

The second row is the sixteen characters `F23C448128E09200` written out one byte each. The third is the same sixteen characters in the encoding IBM mainframes use, where the digit `0` is `0xF0` rather than `0x30`.

Now the failure. Suppose a parser expects raw binary and receives the ASCII hexadecimal form. It reads the first eight bytes, `46 32 33 43 34 34 38 31`, as a bitmap — in binary `01000110 00110010`

`00110011 01000011 00110100 00110100 00111000 00110001` — and declares elements 2, 6, 7, 11, 12, 15, 19, 20, 23, 24, 26, 31, 32, 35, 36, 38, 43, 44, 46, 51, 52, 53, 59, 60 and 64.

Twenty-five elements, including the MAC in element 64. Bit 1 is clear, so the parser concludes there is no secondary bitmap. Nothing about that set is implausible on its face. The parser then reads twenty-five fields out of what are in fact the remaining eight characters of the bitmap followed by the message body, and produces a card number that is not a card number, an amount that is not an amount, and a confident structured object that is entirely fiction. This is why the encoding is the first question on any new link, and why "it parses" is not evidence that it parses correctly.

Fixed and variable length data elements

Every data element has a content type — `a` alphabetic, `n` numeric, `s` special characters, `an`, `as`, `ns` and `ans` for the combinations, `b` binary, `z` for track 2 and track 3 code set per ISO/IEC 7813 and ISO/IEC 4909, `x+n` for a signed amount, `p` for pad — and a length rule.

The length rule is written after the type. A bare number means fixed: `n 6` is exactly six digits, always, and a sender with fewer must pad. Leading dots mean variable, one dot per digit of the length prefix: `n..19` is LLVAR, a two-digit length then up to nineteen digits; `ans...999` is LLLVAR, a three-digit length then up to 999 characters. A single dot, `n.6`, is LVAR, which exists in the notation and is essentially unused. For binary elements the declared number is bits, not bytes, so `b 64` is eight bytes — a constant source of confusion, made worse by tables that use the same notation for character counts.

Most of the catalogue is fixed length, and of the elements that carry a normal card purchase only 2, 32, 35 and 55 are variable. That is deliberate: fixed fields need no length prefix and no bounds arithmetic, and can be copied with a constant offset and a constant count.

LLVAR and LLLVAR on the wire

This is the part of ISO 8583 that produces the most interoperability defects, because the notation `n..19` specifies the semantics of the length prefix and says nothing about its encoding. There are four independent decisions, and each is a place two implementations can differ.

How the length digits are encoded. ASCII digits use one byte per digit, so an LLVAR prefix is two bytes and an LLLVAR prefix three. Packed binary-coded decimal uses one byte per two digits, so an LLVAR prefix is one byte and an LLLVAR prefix two, with a leading zero nibble. The length 27 is `0x32 0x37` in ASCII and `0x27` in BCD. A parser expecting the wrong one is off by one or two bytes at the front of every variable field.

Whether the length is binary or decimal. Some specifications encode the length as a plain unsigned integer rather than decimal digits, making 27 `0x1B`. The pyiso8583 library, which exists to model this variety, treats length encoding as a configuration axis separate from data encoding for this reason.

What the length counts. For character data it counts characters, which is also bytes. For packed BCD numeric data it usually counts *digits*, and the field occupies half that many bytes. For binary data it usually counts bytes. Get this wrong on a packed field and you read half the field or twice it. Serious libraries make it explicit: pyiso8583 offers a `len_count` setting whose values are `bytes` and `nibbles`, with bytes as the default, alongside left- and right-padding options for odd-length packed data. That such an option has to exist tells you how much disagreement is out there.

What happens at the maximum. `n..19` means up to nineteen. It does not mean a sender may send twenty. Scheme specifications routinely tighten the maxima below the ISO values, so a message that is valid ISO 8583 may still be rejected.

LLVAR and LLLVAR are therefore not one thing but a family of closely related things, and the specification that governs your link — not the ISO document, not a reference website — is the only authority on which member of the family you are speaking.

The parser loop

Once the encodings are settled, the entire receive path is about twenty lines.

```
read MTI (4 characters)
read primary bitmap (8 bytes, or 16 characters if hex-encoded)
if bit 1 set:
    read secondary bitmap (same size)
for n from 2 to 128:
    if bit n is not set: continue
    look up definition[n]
    if definition[n] is fixed:
        take definition[n].length units
    else:
        take definition[n].prefix_digits, decode as length L
        take L units
    store as element n
assert no bytes remain
```

There is no tokeniser, no state machine, no backtracking, no recursion, no dynamic allocation if the buffers are pre-sized, and no ambiguity about whether a value is a string or a number, because the definition already said. That is why a terminal with sixteen kilobytes of RAM could do it in 1987 and a switch handling tens of thousands of messages per second can do it now without a garbage collector running.

The last line is the only error detection the format gives you for free, and it is worth more than it looks. If the bitmap and the body agree, the parse consumes exactly the message and stops; if they do not, there are bytes left over or the parser runs off the end. Implementations that silently tolerate trailing bytes throw away the one cheap integrity check they had.

Why sixty-four bits is close to optimal

The bitmap solves a narrow problem: communicate an arbitrary subset of sixty-four known items, as compactly as possible, decodable in constant time by a machine with no memory to spare. It is worth being precise about how good the answer is.

The worst case is exactly optimal. There are 2 to the power 64 possible subsets of sixty-four elements, so distinguishing all of them requires at least 64 bits by counting alone. The bitmap uses exactly 64. No encoding that can express every subset is smaller in the worst case, and the bitmap meets the bound with equality. That is not “close to optimal”, it is optimal, for the problem as stated.

The typical case gives up about a quarter. Real authorisations carry around seventeen to twenty-one elements, not arbitrary subsets. The information content of an arbitrary seventeen-of-sixty-four subset is the base-two logarithm of sixty-four choose seventeen, which is 50.3 bits, or 6.3 bytes. The bitmap spends 8. The overhead against a cardinality-aware ideal coder is about 27 per cent, or 1.7 bytes, and you would pay for that saving with an arithmetic coder in the hot path of every transaction on earth.

The obvious alternative loses above nine fields. The intuitive design is to send a list of the field numbers present, and a field number in the range 1 to 128 fits in seven bits. Seven bits per present field beats a 64-bit bitmap only below about nine fields: at nine the list costs 63 bits and the bitmap costs 64, and from ten onward the bitmap wins and never stops winning. Add a one-byte count so the receiver knows when the list ends and the crossover moves to exactly eight. A card authorisation

carrying twenty-one elements would need 147 bits, nineteen bytes, as a field list, against eight as a bitmap.

Against a modern self-describing format the gap is about two to one. Protocol Buffers encodes each present field with a varint tag holding the field number and wire type, which is one byte for field numbers up to 15 and two bytes for 16 to 2047. The twenty-one elements of our example would cost 34 bytes of tags against 16 bytes for the two bitmaps, on framing overhead alone.

Against a fixed layout the gap is three orders of magnitude. The other way to avoid labels is to give every field a fixed position whether it is used or not. Summing the maximum lengths of data elements 2 to 64 gives 12,733 characters, before the secondary range. Even excluding the twelve elements defined as `...999` — the additional-data and reserved fields — the remainder is 745 characters. A real authorisation whose two bitmaps and twenty-one populated elements come to 351 bytes, with the bitmaps 4.6 per cent of that, makes a fixed layout not a design but a joke about telephone bills.

And sixty-four is the right number for the machine, not just the message. A 64-bit bitmap is one register on any processor built since the mid-1990s. Testing presence is a single AND. Counting fields is a single population-count instruction. Comparing two messages' field sets is a single XOR. In 1987 this meant two 32-bit words rather than one; the committee did not choose 64 because it was a machine word, but it has aged into being one, which is a large part of why nobody has managed to justify replacing it.

The comparison that actually matters

All of the above is about size, and size is the least interesting axis. The real difference between a bitmap and a tagged format is *skippability*.

In Protocol Buffers, JSON, XML or ASN.1 BER, a receiver that meets a field it has never heard of can skip it, because the wire type or the delimiter says how many bytes to step over. Unknown fields are survivable and forward compatibility is free. In ISO 8583 a receiver that meets a field it has no definition for cannot skip it, because the number of bytes to skip is in the definition it does not have. It cannot parse the rest of the message either. There is no recovery.

This single property explains an enormous amount about how the card industry behaves. It is why adding a field to a card network is a multi-year certification programme rather than a deployment, why schemes publish mandates eighteen months ahead, and why private data lives inside elements 48, 62, 63, 124 and 127 as nested tag-length-value structures — because inside a `...999` wrapper whose length the parser *does* know, tags become skippable again, and the schemes rebuilt forward compatibility one level down where the bitmap could not reach. It is also a large part of the argument for ISO 20022, which is self-describing from top to bottom and pays for it in bytes.

Parsing failure modes

These occur in production, and every one presents as a misaligned parse rather than a clean error. That is the point.

Wrong bitmap encoding. Binary read as ASCII hex or the reverse. Off by eight bytes for the whole message. Diagnosed by checking whether the first sixteen bytes all fall in `30` to `39` and `41` to `46` (ASCII) or `F0` to `F9` and `C1` to `C6` (EBCDIC); if they do, it is a hex-encoded bitmap.

Reversed bit order within bytes. Least-significant-bit-first testing. The declared field set is a plausible-looking but wrong permutation: `F2` gives 1, 2, 3, 4, 7 correctly and 2, 5, 6, 7, 8 incorrectly. Diagnosed by decoding a known-good bitmap such as `7010001102C04804` and checking you get 2, 3, 4, 12, 28, 32, 39, 41, 42, 50, 53, 62.

Bit 1 set with no secondary bitmap sent. The parser consumes the first eight bytes of data element 2 as a secondary bitmap. If element 2 begins with the ASCII characters `16411111`, it reads a secondary bitmap declaring elements 67, 68, 72, 75, 76, 78, 79, 83, 84, 86, 91, 92, 96, 99, 100, 104, 107, 108, 112, 115, 116, 120, 123, 124 and 128 — twenty-five phantom fields, several in private-use space — and tries to read them all out of the transaction body. The most destructive bitmap defect there is, and common in hand-rolled senders.

Bit 1 clear with a secondary bitmap sent. The mirror image. The parser reads the secondary bitmap's eight bytes as the value of the first declared element. Because a secondary bitmap is usually mostly zeroes, the classic symptom is a card number of all zeroes.

A bit set for a field the sender did not write. Every subsequent field shifts left by the missing field's length. Fixed fields absorb the shift silently; the first variable field afterwards reads its length prefix from the middle of somebody else's data, and the parse either overruns the buffer or ends with bytes remaining.

A field written without its bit set. The parser stops early, or reads the unannounced bytes as the next declared field. The trailing-bytes assertion catches it, if the implementation has one.

Length prefix encoding mismatch. ASCII versus BCD on LLVAR and LLLVAR. Off by one or two bytes per variable field, cumulative: four variable fields drift by four to eight bytes by the end.

Length counting mismatch. Digits versus bytes on packed numeric fields. Reads half the field or twice it. The symptom is a PAN of the right length with the wrong digits, or half the expected length.

Scheme-tightened maxima. Valid against the ISO catalogue and rejected by the counterparty. This one at least fails loudly, usually with a format error — response code `30` in the 1987 two-character response code set.

Private element with no local definition. Unrecoverable: the parser cannot determine the length and cannot proceed. The correct behaviour is to reject the whole message rather than guess.

MAC failure standing in for a parse failure. Data elements 64 and 128 carry a message authentication code computed over the message, so any change to the bitmap changes the MAC. A bitmap defect on a MAC-protected link therefore presents first as an authentication failure rather than a parse error, which is confusing until you realise the MAC is doing exactly what it should.

The through-line is that positional encoding has no redundancy. Nothing in the byte stream says "this is the amount". The claim that byte 47 is the amount lives entirely in the bitmap and the field table, and if either is wrong the message will still parse, still validate, and still be wrong. Every mature payments organisation ends up with the same three defences: assert that the parse consumes the message exactly, range-check every field against its declared type before using it, and keep a golden set of known-good messages that new builds must reproduce byte for byte.

The bitmap as a diagnostic instrument

Once you can read a bitmap you get several things for free.

You can classify traffic without parsing it. A production switch can read eight bytes and know whether a message carries chip data, a PIN block or a MAC, and whether it is a stripe transaction — routing, fraud-screening and capacity-planning information available before any field is decoded.

You can diff two messages in one instruction. Exclusive-or the two primary bitmaps and the set bits in the result are exactly the elements that differ in presence. Our `0100` request and a corresponding `0110` response, whose primary bitmap is `703800010E808200`, differ by `8204448026601000`, which expands

to elements 1, 7, 14, 18, 22, 25, 35, 38, 39, 42, 43 and 52. The response has dropped the card data and the terminal context and added elements 38 and 39, the authorisation identification response and the response code. The entire logical difference between a request and its answer, in one exclusive-or.

You can specify conformance precisely. Scheme certification is largely a statement about bitmaps: for this message type, these bits must be set, these must not, these are conditional on those. A test plan is, at bottom, a list of required and forbidden bit patterns, which is why it automates so well. And you can read a log over somebody's shoulder — an engineer who decodes the characters after the MTI on sight can tell in seconds whether a failing message is malformed at the top level or merely carrying a value the issuer did not like, which is usually the first useful question in an incident.

Where the bitmap runs out

The bitmap's limits are structural, not incidental, and they are the whole case for the format that will eventually replace ISO 8583.

There are 128 slots, most spoken for by the ISO catalogue and the rest private-use fields that different counterparties have filled with different things. There is no way to add a 129th concept without either the tertiary bitmap nobody implements or a nested structure inside an existing element. The industry chose nesting, which works, but which means the top-level bitmap no longer describes the message — it describes the outer envelope, and the interesting content is inside element 48 or 55 or 127, indexed by a second-level scheme the bitmap knows nothing about.

There is no versioning. The bitmap does not say which field table it was built against. The MTI's first digit names an edition of the standard, but the standard no longer contains the field definitions — in the third edition they were moved out into a separately maintained register. Two systems can agree on the bitmap and disagree on what element 62 is, and nothing in the message will tell them.

And there is no room for meaning. A bit can say "present". It cannot say "present but unreliable", or "present, populated by a stand-in processor rather than the issuer". Every richer statement becomes a value inside another field, which is how element 48 came to contain dozens of subelements in some scheme specifications.

None of that stops the bitmap working. As of writing, in 2026, the overwhelming majority of card authorisations on earth still begin with a four-digit MTI whose first digit says 1987 and then eight bytes of flags, and they will for years yet. It is a forty-year-old trick that is provably optimal for the problem it was given, and the problem it was given was smaller than the one the industry now has. That gap, not any defect in the bitmap, is what the rest of this volume is about. The next chapter takes the fields the bitmap declares and says what each of them actually holds.

25.98 Common wrong ideas

Wrong: `b 64` in a field table means sixty-four bytes. **Right:** it means sixty-four bits, which is eight bytes, which is sixteen hexadecimal characters, and reading it the other way is a classic first-week mistake.

Wrong: bit 1 declares data element 1. **Right:** bit 1 declares a second bitmap covering elements 65 to 128, and its presence or absence moves the byte offset of every subsequent field in the message.

Wrong: the bitmap is an index, so a parser can jump straight to data element 55. **Right:** it declares presence and order and nothing else; several earlier elements are variable length, so you must measure 2 through 43 before you know where 55 begins.

Wrong: test a bit with `byte & (1 << bit)`, as everywhere else in computing. **Right:** bits are numbered from the most significant bit of the first byte, so element *n* is `0x80 >> ((n - 1) % 8)` in byte `(n - 1) / 8`; the least-significant-first version yields a plausible-looking but wrong set of field numbers.

Wrong: the bitmap is sixteen characters on the wire. **Right:** that is one of three encodings in real use, alongside eight raw binary bytes and sixteen EBCDIC characters, and nothing in the message says which one you are receiving.

Wrong: if the message parses, the parse is correct. **Right:** positional encoding has no redundancy, so a wrong bitmap or a wrong field table produces a complete, valid-looking, entirely fictional object; only asserting that the parse consumes the message exactly catches it.

Wrong: a parser can skip a private field it does not recognise. **Right:** the number of bytes to skip lives in the definition it does not have, so it can neither skip the field nor parse the rest of the message, and the correct behaviour is to reject the whole thing.

Wrong: LLVAR means a two-byte ASCII length prefix. **Right:** `n..19` fixes the semantics of the prefix and says nothing about its encoding, which may be ASCII digits, packed binary-coded decimal or a binary integer, and which may count bytes or nibbles.

Wrong: the tertiary bitmap is how you get past 128 elements. **Right:** it is described everywhere as effectively unused on card rails, and the schemes extended instead by nesting tag-length-value structures inside elements they already had.

Wrong: a message that is valid ISO 8583 will be accepted. **Right:** scheme specifications routinely tighten the maxima below the ISO values, so a valid message can still come back with a format error.

25.99 Chapter summary in 20 lines

1. Every ISO 8583 message begins with eight bytes that say nothing about the transaction and everything about the message: which of its sixty-four possible fields the sender filled in.
2. A set bit means the corresponding data element is present, a clear bit means it is absent, and the declared elements follow in ascending numerical order.
3. Because the bitmap gives both the set of fields and their sequence, the message needs no field names, no separators and no offsets anywhere.
4. Sixty-four bits is sixteen hexadecimal characters, and reading them four bits at a time, worth 8, 4, 2 and 1, is the whole skill.
5. The primary bitmap `F23C448128E09200` declares elements 1, 2, 3, 4, 7, 11, 12, 13, 14, 18, 22, 25, 32, 35, 37, 41, 42, 43, 49, 52 and 55, which is a chip-and-PIN purchase in a shop, read without seeing a single value.
6. Bit 1 is not a data element: it announces a secondary bitmap covering elements 65 to 128, which sits immediately after the primary bitmap and before any value.
7. Getting bit 1 wrong is the highest-blast-radius single-bit error in the format, because it shifts every subsequent field by eight or sixteen bytes.
8. The bitmap says which fields are present; the field table says how long each is, and fixed lengths are simply counted while variable ones carry a two- or three-digit length prefix.
9. The receive path is therefore a loop of about twenty lines with no tokeniser, no state machine, no backtracking and no allocation, which is why a terminal with sixteen kilobytes of memory could run it in 1987.
10. Its one free integrity check is that the parse must consume the message exactly, and implementations that tolerate trailing bytes throw that check away.

11. The same sixty-four bits travel as eight raw binary bytes, sixteen ASCII characters or sixteen EBCDIC characters, and the encoding is the first thing to establish on any new link because the message does not carry it.
12. A parser configured for the wrong encoding is not slightly wrong; it produces a confident, well-formed and entirely fictional transaction from the first byte.
13. Bits are numbered from the most significant bit of the first byte, and the usual least-significant-first instinct yields a plausible but wrong field set that looks like field numbers either way.
14. For the problem as stated the bitmap is exactly optimal in the worst case, costs about 27 per cent more than an ideal cardinality-aware coder on real traffic, and beats a list of field numbers from roughly the tenth field onward.
15. Against Protocol Buffers the framing gap is about two to one, and against a fixed layout with every field always present it is three orders of magnitude.
16. The property that actually matters is not size but skippability: a tagged format can step over a field it has never heard of, and ISO 8583 cannot, because the length is in the definition the receiver lacks.
17. That single property explains multi-year certification programmes, mandates published eighteen months ahead, and the whole practice of nesting private data inside elements 48, 62, 63, 124 and 127 where lengths are known.
18. Every bitmap defect presents as a misaligned parse rather than a clean error, with symptoms such as a card number of all zeroes, an amount with extra digits, or a MAC failure standing in for a parse failure.
19. Once read fluently the bitmap becomes a diagnostic instrument: it classifies traffic before any field is decoded, and an exclusive-or of two bitmaps names exactly the elements by which a request and its response differ.
20. The bitmap's limits are structural — 128 slots, no versioning, and no way to say anything richer than "present" — and that gap, rather than any defect in the design, is the case for whatever eventually replaces it.

Chapter sources: ISO 8583:2023, third edition, July 2023, on the message bitmap component, via the publicly available standard preview; the ISO catalogue entry for ISO 8583-1:2003; the Wikipedia ISO 8583 article for the 1987 data element catalogue, the format and length notation tables, the worked bitmap 7010001102C04804 and the 1987 response code table; the ISOParser ISO 8583 reference for the field catalogue with LLVAR and LLLVAR annotations, the worked bitmap 7234054128C08000, the note that binary element lengths are expressed in bits, and Visa BASE I/BASE II and Mastercard IPM private-field practice; the pyiso8583 v4.0.1 specifications manual for length encoding, length-count in bytes versus nibbles and odd-length packed padding; the moov-io/iso8583 bitmap documentation for bitmap sizing; and the IBM App Connect ISO8583 sample documentation for the rarity of the tertiary bitmap. All bitmap expansions, byte counts, encoding tables and size, crossover and information-content calculations were computed from those definitions and verified against the published worked examples.

4

CHAPTER

The Fields That Matter

26.0 What this chapter gives you

1. You will be able to say, for each of the fifteen fields that carry a card authorisation, what it holds, who writes it and what specifically breaks when it is wrong.
2. You will be able to read a field definition such as `n..19`, `ans 8` or `ans...999` and know the character set, the length rule, and the justification and padding that go with it.
3. You will be able to explain why data element 4 carries `000000004250` with no decimal point, and why the position of that point is determined by data element 49 rather than by the amount.
4. You will be able to decompose a processing code such as `010300` into transaction type and two account types, and say why an account-type pair of `00` does not mean "current account".
5. You will be able to say which clock a scheme rule is talking about when it names DE7, DE12, DE13 or DE15, and why a purchase at 23:58 on 31 December can be reconstructed 364 days wrong.
6. You will be able to explain why sending `01` in DE22 for a card that was physically present is a commercial event rather than a technical one.
7. You will be able to distinguish the identifiers by lifetime — the STAN that lasts one request-and-response pair, the RRN that lasts into a chargeback weeks later — and say which one an acquirer will ask a merchant for.
8. You will be able to state exactly which parts of an authorisation are cryptographically protected and which are unauthenticated assertions made by whoever is sending.
9. You will be able to name which tags inside DE55 come from the card and which from the terminal, and say why the amount, currency and country have to appear twice in one message.
10. You will be able to explain why DE39 `00` does not mean the transaction completed, and what else a response can carry that changes the outcome.

The previous two chapters gave you the envelope and the index. The message type indicator says what kind of message this is; the bitmap says which numbered boxes are inside. Neither says anything about the contents, and the contents are where integrations actually fail.

This is worth stating plainly. A message that will not parse is an easy problem: it fails immediately, it fails identically every time, and the fix is usually one line. The expensive failures are the ones where every field parses perfectly, the bitmap is correct, the lengths are right, and the value in one box is simply wrong — a refund flagged as a purchase, an amount off by a factor of a hundred, a terminal identifier reused across two shops, a reference number regenerated on a retry. Those do not surface at the parser. They surface three days later in a reconciliation report, or six weeks later in a chargeback, and by then the message that caused them has been through four systems belonging to three companies.

So this chapter walks the fields one at a time. For each one: what it holds, who writes it, and what specifically goes wrong when it is wrong. There are 128 numbered elements in ISO 8583 and 192 if you count the tertiary bitmap, but a working card authorisation is built from perhaps twenty, and about fifteen of those carry the weight. Those fifteen are the subject here.

The plain version

Imagine a form that travels along a row of desks.

The form has numbered boxes and no labels — that trick you already know. What you do not yet know is the rule that governs the desks. Every desk along the row is allowed to write in *some* of the boxes, and only those. Every other box on the form it must copy across to the next desk exactly as it received it: same characters, same spaces, same everything. Copy it wrong and nobody notices at your desk. They notice at the far end, or on the way back, when the form no longer matches the one that was sent.

There are four desks. The till in the shop. The shop's bank. The card company in the middle. Your bank. The form goes left to right, your bank writes the answer into two boxes, and the form comes back right to left.

Who writes what

The till fills in almost everything factual about the sale. It knows the card number, because the card just told it. It knows the amount, because the shopkeeper just typed it. It knows what kind of sale this is — a purchase, not a refund. It knows what time its own clock says. It knows how the card was presented: tapped, inserted, swiped or typed in by hand. It knows which till it is and which shop it belongs to. And it knows what the chip said, because the chip signed a little package of data and handed it over.

The shop's bank fills in the boxes about the journey rather than the sale. What time it is sending this message. A serial number of its own so it can match the answer to the question. Its own identifying number. And a reference number that it invents now and that will follow this transaction for the rest of its life — into the shop's bank statement, into the settlement file, and into any argument that happens later.

The card company in the middle mostly reads. It looks at the first six or eight digits of the card number, works out which bank issued it, and forwards the form.

Your bank writes exactly two boxes. Box 38 gets a six-character approval code that your bank makes up on the spot — the thing printed on the receipt as "AUTH CODE". Box 39 gets two characters that mean yes or no. Everything else on the returning form is a copy.

The worked example

The same purchase as two chapters ago. A book, £42.50, at a bookshop in London, on 17 March 2026, at 32 minutes and 5 seconds past two in the afternoon. Here is the form when it leaves your bank on the way back, with the author of each box named.

Box	Contents	Who wrote it
2	4111111111111881	The till, from the card
3	000000 (a purchase)	The till
4	000000004250	The till
7	0317143205	The shop's bank
11	004417	The shop's bank

Box	Contents	Who wrote it
12	143205	The till's clock
13	0317	The till's clock
22	071 (tapped, has a PIN pad)	The till
37	607614430041	The shop's bank
38	A4T29B	Your bank
39	00 (yes)	Your bank
41	TERM0042	The till
42	KEDBYTEBOOKS01	The till
49	826 (pounds)	The till
55	(the chip's signed package)	The chip and the till together

Fifteen boxes. Two of them written by the bank that decided; thirteen of them written before the message ever left the shop.

Three ways this goes wrong

The amount with no decimal point. Box 4 is `000000004250`. There is no decimal point in it and there never will be, because box 49 says `826` and everybody has agreed that pounds have two digits after the point. Now suppose the till was set up by someone who thought box 4 held whole pounds. It sends `000000000042`. The customer is charged 42 pence, the shop is short by £42.08, and nothing anywhere reports an error, because `000000000042` is a perfectly valid amount. Or suppose the mistake runs the other way and the till sends `000000425000`. The customer is charged £4,250 for a book, and the first anyone hears about it is a telephone call.

Two tills with the same serial number. Box 11 is a serial number the shop's bank uses to match answers to questions. Suppose the shop opens a second till and the engineer copies the configuration from the first, including the counter it starts from. Now two purchases, seconds apart, both go out as number `004417`. Two answers come back. Each till takes the first answer that arrives. There is a fifty-fifty chance that till one prints the approval that belonged to till two, and the customer whose card was actually declined walks out with a book.

The box that says how the card was read. Box 22 says `071`: the card was tapped, on a terminal that has a PIN pad. Suppose the till is configured to send `011` instead, which means the number was typed in by hand. Nothing fails. The purchase is approved and the receipt prints. But "typed in by hand" is what a telephone order looks like, and the rules that decide who pays when a purchase turns out to be fraud treat a typed-in number very differently from a tapped card. The shop has just given away its protection, at a slightly higher fee, by sending one digit it did not mean.

None of those three is a software crash. All three are one box holding a value that is the wrong kind of true.

Where the plain version stops being true

The desks do not merely copy. The rule that every desk copies other people's boxes unchanged is the single most useful lie in the plain version, and it is a lie. Acquirers routinely rewrite fields on the way through. They renumber the serial number in DE11, because the number the terminal used is unique only within that terminal and the acquirer needs one unique within itself. They may replace the card number in DE2 with a network token, so the number the issuer sees is not the number embossed on the card. They translate processing codes between their own terminal interface and the scheme's

dialect. They convert amounts when the settlement currency differs from the transaction currency, adding DE5 and DE50 that the terminal never sent. The practical consequence is that the message the issuer receives is not the message the terminal sent, and when you are debugging across an acquirer boundary you are looking at two different messages that describe one purchase. Field values are not conserved along the chain. Only a handful are, and knowing exactly which is most of the skill.

One field does not have one meaning. ISO 8583 fixes the number, the name and the format of each data element. It does not fix the vocabulary inside most of them. DE3's account-type pairs, DE22's entry-mode codes, DE39's response values and DE55's tag requirements are all set by the scheme, and the schemes disagree with each other in small, specific, expensive ways. The same six digits in DE3 can mean a cash advance from a credit account on one network and something the next network does not implement at all. Worse, the field's own shape changes between editions of the standard: DE22 is three digits in the 1987 edition and a twelve-position code in 1993; DE39 is two characters in 1987 and three in 1993; DE12 is a local time in 1987 and a local date-and-time in 1993. "Which field" is never a complete question. The complete question is which field, on which scheme, in which edition, in which direction.

The reply is not two boxes. The plain version says the issuer writes DE38 and DE39 and copies the rest. In practice the response can also carry a different amount from the one requested, when the issuer approves a partial amount and the terminal must handle the shortfall. It can carry balances in DE54. It can carry issuer authentication data and issuer scripts in DE55, which the terminal must hand back to the chip, and which can cause a card to refuse a transaction that DE39 just approved. It can carry additional response data in DE44 and private data in DE48 that changes what the terminal is required to display. Treating DE39 as the whole answer is how terminals end up printing "APPROVED" for transactions that did not complete.

Almost nothing in the message is proof. The last correction is the one that reframes everything else. Of the fifteen fields in the table above, exactly one is cryptographically protected against a lying sender: DE55, which contains a cryptogram the chip computed with a key only the chip and the issuer hold. DE52 protects the PIN. DE64 protects the message as a whole between two adjacent parties. Everything else — the entry mode, the terminal identifier, the merchant identifier, the amount, the currency — is an unauthenticated assertion made by whoever is sending. The issuer cannot verify that DE22 is honest. It can only decide how much to trust the party that sent it. This is why card payments are governed by liability rules and scheme membership rather than by proof, and it is why an incorrect DE22 is a commercial event rather than a technical one.

The technical version

Reading a field definition

Every scheme specification describes a data element with the same shorthand, and it is worth being precise about it before using it fifteen times.

The letters give the character set. `n` is numeric, digits only, and is conventionally carried as packed binary-coded decimal, two digits to the byte. `a` is alphabetic, `an` alphanumeric, `ans` alphanumeric plus the special characters, `b` binary, and `z` the track-2 character set of ISO/IEC 7813, which includes the field separator. `x+n` denotes an amount preceded by a debit or credit indicator.

The number gives the length, and the notation for the length is the part people get wrong. `n 6` is fixed: exactly six digits, always present in full. `n..19` is variable with a two-digit length prefix, conventionally written LLVAR: the field begins with two digits saying how many follow, so a sixteen-digit card number

is transmitted as `16` then the sixteen digits. `ans...999` is variable with a three-digit length prefix, LLLVAR. Some specifications write `n..19` as `n-19` or `LLVAR n 19`; they mean the same thing.

Fixed numeric fields are right-justified and zero-filled. Fixed alphanumeric fields are left-justified and space-filled. That asymmetry causes real defects: a terminal identifier of `TERM42` in an eight-character field is `TERM42` plus two spaces, not `00TERM42`, and a system that trims trailing spaces before comparing will match records that a system which does not trim will treat as different.

DE2, primary account number

Format `n..19`, LLVAR, up to nineteen digits.

DE2 carries the account number as defined by ISO/IEC 7812-1: an issuer identification number, an individual account identifier, and a Luhn check digit in the final position. Real lengths run from twelve to nineteen, with sixteen dominant and fifteen normal for American Express.

It is populated by the terminal, from track 2 of the magnetic stripe, from EMV tag `5A` on a chip card, or by hand from the embossing. It is frequently *re*populated afterwards. Under network tokenisation the value the issuer sees is a device token or a merchant token rather than the funding account number, and the mapping back to the real number happens inside the scheme. Under point-to-point encryption the acquirer's decryption endpoint puts the clear PAN into DE2 for the first time at the edge of its own secure zone.

DE2 is also, quite often, absent. When the terminal sends full track 2 data in DE35, the account number is already inside that field, and many acquirer interfaces treat DE2 and DE35 as mutually exclusive rather than duplicating the number. A parser that assumes DE2 is always present will fail on perfectly valid magnetic-stripe traffic.

What goes wrong. Odd-length PANs in packed BCD need a padding nibble, and implementations differ on whether it goes at the front or the back; get it wrong and a nineteen-digit number arrives with a stray zero. Truncation at eighteen digits is a recurring defect in systems written when sixteen was universal. Storing DE2 anywhere it does not need to be is the single most common PCI DSS finding in card systems, and the field's presence in logs is the usual culprit. Finally, a mismatch between DE2 and the account number embedded in DE35 or in EMV tag `5A` is a hard decline at most issuers, because it is exactly what a tampered message looks like.

DE3, processing code

Format `n 6`, fixed, six digits in three pairs.

Positions 1 and 2 are the transaction type. Positions 3 and 4 are the account the money comes from. Positions 5 and 6 are the account it goes to. It is populated by the terminal and frequently translated by the acquirer.

The transaction types that matter in practice, as documented for Mastercard and Visa authorisation traffic:

Code	Transaction type
00	Goods or service purchase, point of sale only
01	Withdrawal or cash advance
09	Purchase with cash back
10	Account-funding transaction, incoming
11	Quasi-cash
17	Cash disbursement

Code	Transaction type
18	Card account verification
20	Purchase return or refund
21	Deposit
28	Payment transaction, incoming Mastercard OCT
26	Money transfer, incoming Visa OCT
30	Balance enquiry
40	Account transfer
50	Bill payment or payment to a third party

The account-type pairs are a much shorter list: `00` default, `09` other, `10` savings, `20` checking, `30` credit.

So `000000` is a purchase from the default account, `010100` a cash withdrawal from savings, `010300` a cash advance against credit, `200000` a refund, `300000` a balance enquiry.

The divergence between networks is public and it is not subtle. Not every network implements every transaction type, and several codes carry network-specific meaning: `17` marks the over-the-counter cash disbursement flag on Mastercard, while the equivalent condition on Visa is `01` combined with merchant category code 6010. A processing code is meaningful only against a named specification.

What goes wrong. Sending `000000` for what is commercially a refund produces a second debit rather than a credit, and the fix involves a manual adjustment and an unhappy customer. Sending a balance enquiry with a non-zero DE4 is rejected by some issuers and silently treated as a purchase attempt by others. Specifying an account type the cardholder does not have produces DE39 `52` or `53` — "no checking account", "no savings account" — which support teams reliably misdiagnose as an issuer outage, because the card works fine everywhere else. And an account-type pair of `00` does not mean "current account". It means "not specified, issuer decides", which is a different thing and behaves differently at an ATM.

DE4, amount, transaction

Format `n 12`, fixed, twelve digits, unsigned, no decimal separator.

The value is expressed in the minor units of the currency named in DE49. The decimal position is implied by the ISO 4217 exponent for that currency and is never transmitted. Two decimal digits for sterling (`826`) and US dollars (`840`); none for Japanese yen (`392`); three for Kuwaiti dinar (`414`). The maximum representable value is 999,999,999,999 minor units.

It is populated by the terminal. It is also, on occasion, populated by the issuer: a partial approval returns an amount lower than the amount requested, and the terminal must then either collect the difference by another means or reverse the whole thing.

DE4 does not travel alone. DE5 is the settlement amount, in the currency the acquirer settles in, added by the acquirer or the scheme. DE6 is the cardholder billing amount, in the currency of the cardholder's account, added by the issuer side. DE54 carries additional amounts, each with its own account type, amount type, currency code and sign, which is how ATM balances and cashback components travel. Four fields, four currencies, four opportunities for a conversion to be applied twice.

What goes wrong. Floating-point arithmetic anywhere near this field is a defect waiting for a specific input; amounts belong in integers of minor units or in decimal types with explicit scale. Hard-coding two decimal places breaks the moment the system sees yen or dinar. Estimated amounts create their own class of problem: a fuel pump authorises a nominal amount and clears the real one, a hotel

authorises a nightly estimate and increments it, and every one of those flows depends on the clearing amount being allowed to differ from the authorised amount within scheme-defined tolerances. And an account verification is DE3 transaction type `18` with DE4 set to zero, not a one-pound purchase followed by a reversal, although the industry spent a decade doing the latter.

DE7, DE12 and DE13, the three clocks

DE7 is transmission date and time, `n 10`, MMDDhhmmss, in Coordinated Universal Time, set by the sender of the individual message. It changes at every hop. It is not the time of the purchase; it is the time this particular leg was transmitted.

DE12 is local transaction time, `n 6`, hhmmss, in the local time of the point of sale, set by the terminal. DE13 is local transaction date, `n 4`, MMDD, likewise.

Together they are the reason chapter after chapter of scheme documentation insists on which clock a rule refers to. A cut-off expressed in DE7 is a different instant from one expressed in DE12 and DE13.

The edition trap is severe here. In the 1987 edition, DE12 is a time and DE13 is a date, and neither carries a year. In the 1993 edition DE12 becomes a combined local transaction date and time, twelve digits, and DE13 is redefined. A system that has to reconstruct a full timestamp from 1987-edition fields must supply the century and year from context, and the last days of December are where that inference fails: a transaction at 23:58 local on 31 December, transmitted at 00:02 UTC on 1 January, produces a DE7 whose date is in the following year and a DE13 that is not, and a naive reconstruction dates the purchase 364 days wrong.

What goes wrong, beyond the year. Terminal clocks drift, and nothing in the protocol corrects them; a terminal an hour out because nobody handled British Summer Time will put transactions in the wrong reconciliation window even though DE7 is perfect. Systems that use DE7 for duplicate detection and DE12 for reporting will produce two internally consistent reports that disagree with each other. And DE15, the settlement date, is a third date again, owned by the acquirer, and is the one that actually determines when money moves.

DE11, system trace audit number

Format `n 6`, fixed, six digits.

The STAN is assigned by the institution that originates the message. It is unique within that originator, within a processing day, and it wraps at 999999 back to 000001. It is not globally unique, it was never intended to be, and treating it as a transaction identifier is a design error that surfaces at scale.

Its purpose is narrow and important: it is part of the key that lets two adjacent systems match a response to a request. The full matching key is set by each scheme rather than by ISO, but it conventionally combines DE11 with DE7, DE32 (the acquiring institution identification code) and DE37.

The rules around it are strict in three places. A repeat message — origin digit changed from 0 to 1, producing `0101` from `0100` — must carry the same STAN as the original, because the whole point is for the receiver to recognise it as the same transaction rather than a new one. A reversal quotes the original STAN inside DE90. And an advice must not reuse a STAN that is still in the matching window for a different transaction.

What goes wrong. Two terminals sharing a STAN sequence, described in the plain version, is the classic. So is a STAN counter that resets on reboot rather than persisting, which produces collisions in a burst after every power cut. So is generating a fresh STAN on a retransmission, which converts

a timeout into a genuine double charge, because the issuer now sees two unrelated authorisation requests instead of one request and its repeat.

DE22, point of service entry mode

In the 1987 edition, `n 3`: two digits of PAN entry mode followed by one digit of PIN entry capability. It is set by the terminal, and it is the field that decides a large part of the commercial outcome of the transaction.

The PAN entry mode values, as documented against the Mastercard Customer Interface Specification and VisaNet authorisation messages:

Code	PAN entry mode
00	Entry mode unknown
01	Manual entry
02	Magnetic stripe
03	Bar code reader
04	Optical character reader
05	Integrated circuit card, contact chip
07	Contactless chip
09	Electronic commerce with a DSRP cryptogram in DE55 (Mastercard)
10	Electronic commerce, credential on file
79	Hybrid terminal failed to send a fallback transaction or failed to read the chip (Mastercard)
80	Chip failed at a chip-capable terminal, defaulted to magnetic stripe read
81	Electronic commerce with an authentication value in the UCAF (Mastercard)
82	Auto PAN entry via server
90	Magnetic stripe, full unaltered track data present
91	Contactless magnetic stripe, full unaltered track data present
95	Chip card with unreliable card verification value data (Visa only)

The third digit, PIN entry capability, is a much shorter list: `0` unspecified, `1` the terminal has PIN entry capability, `2` it does not, `3` software-based PIN entry on mobile, `8` the terminal has a PIN pad but it is not currently operative.

Note the pairs that are easy to confuse and expensive to confuse. `05` and `07` are chip transactions, contact and contactless, and both carry a cryptogram in DE55. `02`, `90` and `91` are magnetic-stripe transactions, and `91` is a contactless tap that produced stripe-equivalent data rather than a chip cryptogram — a different security posture entirely from `07` despite looking identical to the customer. `80` is chip fallback, and many issuers decline it by standing policy because it is the signature of a cloned card being presented at a terminal whose chip reader has been deliberately damaged.

The 1993 edition replaces all of this. DE22 becomes the point of service data code, twelve positions rather than three, each describing one property of the acceptance environment. American Express's Global Credit Authorization Guide, which is written to ISO 8583:1993, lays them out as: card data input

capability, cardholder authentication capability, card capture capability, operating environment, cardholder present, card present, card data input mode, cardmember authentication method, cardmember authentication entity, card data output capability, terminal output capability and PIN capture capability. Mastercard carries substantially the same information in DE61 in its 1987-format authorisation messages. Same facts, three different homes, depending on which specification you are implementing.

What goes wrong. Everything downstream of DE22 is a rule about liability and price. Sending `01` for a card that was physically present forfeits card-present interchange and card-present chargeback protection. Sending `07` when the terminal actually performed a magnetic-stripe read causes the issuer to look for a cryptogram in DE55 that is not there, producing a decline that looks like a network fault. Sending `05` while omitting DE55 entirely is the same failure in the other direction and is one of the most common findings in scheme certification testing. And the PIN entry capability digit must match reality: a terminal that claims capability `1` and then never sends a PIN block in DE52 will have its cardholder verification method results in DE55 questioned by the issuer.

DE37, retrieval reference number

Format `an 12`, fixed, twelve alphanumeric characters.

The RRN is assigned by the acquirer and is echoed unchanged by everyone downstream. It is the identifier with the longest life in the whole message: it persists from authorisation into the clearing record, into the settlement report, and into the retrieval requests and chargebacks that may follow weeks later. When a merchant telephones an acquirer about a specific transaction, the RRN is what the acquirer wants.

Its internal construction is not fixed by ISO 8583, which defines it only as a reference supplied by the system retaining the original source information. The schemes impose their own conventions. A Julian date prefix in the form YDDD — one digit of year, three digits of day of year — is widespread; Mastercard's IPM clearing format defines the first four positions of DE37 as the Julian processing date the acquirer assigned to the transaction. A common full convention extends that with a two-digit hour and a six-digit sequence, which is how the example `607614430041` decomposes: year 6, day 076 which is 17 March 2026, hour 14, sequence 430041.

What goes wrong. Regenerating the RRN on a retransmission destroys the receiver's ability to recognise the retransmission, with the same consequence as a regenerated STAN. Failing to echo it exactly — trimming trailing spaces, upper-casing, treating it as a number and dropping leading zeros — breaks reversal matching, because the reversal will not join to the original. Reusing RRNs within the retention window creates records that cannot be told apart in a dispute, and the party that cannot produce a unique reference is the party that loses the dispute.

DE38, authorisation identification response

Format `an 6`, fixed, six alphanumeric characters.

Generated by the issuer, returned in the response, printed on the receipt as the authorisation code, and required in the subsequent clearing record. It is the issuer's own handle on the approval it granted.

What goes wrong. Acquirers and gateways occasionally invent a value for DE38 when the issuer did not return one — for offline approvals, for stand-in processing, or simply as a placeholder. That value will not match anything at the issuer, and the clearing record carrying it is a candidate for a chargeback on the grounds of no valid authorisation. The reverse also happens: an approval in DE39 with DE38 blank or space-filled, which some acquirer platforms accept and then cannot present. And the field is

alphanumeric, so a system that stores it in a numeric column will corrupt any code containing a letter, which is most of them.

DE39, response code

Format `an 2` in the 1987 edition, `an 3` in 1993 and later. Set by the issuer, or by the scheme when the scheme answers on the issuer's behalf.

The 1987 vocabulary is a flat two-character list. The values that carry real operational weight are these:

Code	Meaning
00	Approved or completed successfully
01	Refer to card issuer
03	Invalid merchant
04	Pick-up
05	Do not honour
12	Invalid transaction
13	Invalid amount
14	Invalid card number, no such number
30	Format error
41	Lost card
43	Stolen card, pick-up
51	Not sufficient funds
54	Expired card
55	Incorrect personal identification number
57	Transaction not permitted to cardholder
58	Transaction not permitted to terminal
61	Exceeds withdrawal amount limit
62	Restricted card
65	Exceeds withdrawal frequency limit
91	Issuer or switch is inoperative
96	System malfunction

Codes `04`, `07`, `41` and `43` carry card-retention semantics: the terminal is being instructed to capture the card, which unattended terminals implement literally and countertop terminals implement as a message to the operator.

The 1993 edition replaces the flat list with a three-digit structure in which the leading digit identifies the class of outcome, so `000` is approved, `100` is a plain decline, `200` is a decline with card capture, `9xx` is could-not-process. The two systems coexist. A single transaction can pass through an interface using the 1987 codes and another using the 1993 codes, and the mapping between them is lossy in both directions. Response codes get a chapter of their own later in this volume, because the interesting question about them is not what they are but why so many of them are deliberately uninformative.

What goes wrong here is chiefly interpretation. DE39 `00` does not mean the transaction completed; it means the issuer approved it. The terminal can still fail, the chip can still decline in the second generate-AC after examining the issuer's response in DE55, and the customer can still walk away. A system that treats `00` as terminal state rather than as one input to terminal state will under-report reversals and over-report revenue.

DE41 and DE42, terminal and merchant identity

DE41, card acceptor terminal identification, is `ans 8`, fixed, left-justified and space-filled. DE42, card acceptor identification code, is `ans 15`, fixed, left-justified and space-filled. DE43, card acceptor name and location, is `ans 40` in the 1987 edition.

DE41 is assigned by the acquirer and configured into the terminal. It must identify one physical acceptance point uniquely within the card acceptor. DE42 is the acquirer's identifier for the merchant, and it is the field on which the acquirer's settlement, its interchange qualification and its risk monitoring all hang. DE43 is the human-readable name and town, and it is the raw material from which the descriptor on the cardholder's statement is built.

What goes wrong. Duplicate DE41 values across sites — the same terminal identifier deployed to two shops after a hardware swap — make it impossible to attribute a transaction to a location, which matters both for the merchant's own reporting and for fraud investigation. A mismatch between the DE42 in the authorisation and the DE42 in the clearing record breaks the acquirer's matching and produces unmatched clearing, which is expensive to resolve manually. And DE43 is quietly one of the highest-leverage fields in the entire message: a merchant whose descriptor reads as an unrecognisable holding-company name generates "I do not recognise this transaction" disputes at a multiple of the rate of one whose descriptor reads as the shop's trading name and town. That is a chargeback cost caused entirely by forty characters of free text.

DE49, currency code, transaction

Format `n 3` in general use, defined in the 1987 edition as three characters which may be alphabetic or numeric. In practice every major scheme requires the ISO 4217 numeric code: `826` sterling, `840` US dollar, `978` euro, `392` yen, `356` Indian rupee.

It is set by the terminal, from its own configuration rather than from the card. It says what currency DE4 is denominated in, and by implication where the decimal point sits, because the ISO 4217 minor-unit exponent is not carried in the message.

Its companions are DE50, the settlement currency, and DE51, the cardholder billing currency. A cross-border purchase can legitimately involve three different currency codes in one message chain, and dynamic currency conversion adds a fourth relationship, because the cardholder is offered a rate at the terminal and the transaction is then presented in the cardholder's own currency with the conversion already applied.

What goes wrong. Sending the alphabetic code `GBP` where the interface expects `826` is a format error, and it is a common first-day defect because the alphabetic form is what humans write. Defaulting to `840` in a terminal configuration template is worse, because it does not fail: it produces a valid message denominated in a currency the shop does not trade in. Assuming two decimal places universally corrupts every yen and every dinar amount by two orders of magnitude in one direction or three in the other. And a currency code that differs between the authorisation and the clearing record will not match, which the acquirer will see as an unmatched presentment.

DE55, integrated circuit card data

Format `ans...999` in the 2023 edition of the standard; schemes cap it far lower, commonly at 255 bytes with a three-digit length prefix. It is a format inside a format, and it is parsed by entirely different rules from the message carrying it.

The 2023 standard is explicit that its own composite-dataset structure does not apply to the ICC-related data element, because the linking of related sub-elements there follows ISO/IEC 7816-6. DE55 is there-

fore a sequence of EMV tag-length-value objects, with tags of one or two bytes, lengths in BER-TLV form, and values whose interpretation is defined in EMV Book 3 rather than in ISO 8583 at all.

The tags that carry the transaction, on the way to the issuer:

Tag	Name	Origin
9F26	Application cryptogram	Card
9F27	Cryptogram information data	Card
9F10	Issuer application data	Card
9F36	Application transaction counter	Card
82	Application interchange profile	Card
5F34	Application PAN sequence number	Card
9F09	Application version number	Card
95	Terminal verification results	Terminal
9A	Transaction date	Terminal
9C	Transaction type	Terminal
9F02	Amount, authorised	Terminal
9F03	Amount, other	Terminal
5F2A	Transaction currency code	Terminal
9F1A	Terminal country code	Terminal
9F33	Terminal capabilities	Terminal
9F34	Cardholder verification method results	Terminal
9F35	Terminal type	Terminal
9F37	Unpredictable number	Terminal
9F41	Transaction sequence counter	Terminal
84	Dedicated file name	Card

And on the way back, from the issuer:

Tag	Name	Origin
91	Issuer authentication data	Issuer
71	Issuer script template 1	Issuer
72	Issuer script template 2	Issuer

The division of labour is the point. The card supplies what only the card knows and signs it; the terminal supplies the context the card was asked to sign over; the issuer recomputes the cryptogram from the same inputs using the same key and either gets the same answer or does not. The reason 9F02, 5F2A, 9F1A and 9F37 must appear in DE55 even though the amount, the currency and the country are already elsewhere in the message is that these are the values the card signed. If DE4 says one amount and tag 9F02 says another, the cryptogram will validate against 9F02 and the issuer now knows that something between the chip and it altered the amount.

What goes wrong. Truncation at a scheme's byte limit silently removes trailing tags, and if 9F10 is one of them the issuer cannot derive the session key and cannot validate the cryptogram, producing a decline with no informative code. Sending the field as ASCII hexadecimal where the interface expects binary, or the reverse, doubles or halves every length and yields a parse failure at the first tag. Re-ordering tags is permitted by BER-TLV and is nonetheless a frequent cause of failure in implementations that parse by position rather than by tag. Constructed tags — templates whose value is itself a sequence of TLVs — need recursive parsing, and a flat parser that meets tag 71 will read an issuer

script as a single opaque blob. And omitting DE55 entirely on a transaction whose DE22 claims `05` or `07` is the certification failure named earlier: the message says a chip was used and contains no evidence that one was.

The fields in one table

DE	Name	Format	Written by	Echoed unchanged	Principal failure
2	Primary account number	n..19 LLVAR	Terminal, may be tokenised	Usually	Truncation, BCD padding, mismatch with DE35
3	Processing code	n 6	Terminal, may be translated	Yes	Refund sent as purchase; wrong account type
4	Amount, transaction	n 12	Terminal; issuer on partial approval	Usually	Wrong implied exponent; floating point
7	Transmission date and time	n 10	Each sender, UTC	No, changes per hop	Confused with transaction time
11	System trace audit number	n 6	Message originator	Within a leg	Collision; regenerated on retry
12	Local transaction time	n 6 (1987)	Terminal clock	Yes	Clock drift; no year
13	Local transaction date	n 4 (1987)	Terminal clock	Yes	Year-end rollover
22	Point of service entry mode	n 3 (1987)	Terminal	Yes	Misstated entry mode shifts liability
37	Retrieval reference number	an 12	Acquirer	Yes, exactly	Regenerated or reformatted; disputes fail
38	Authorisation identification response	an 6	Issuer	Into clearing	Invented locally; lost in numeric storage
39	Response code	an 2 (1987), an 3 (1993)	Issuer or scheme	Response only	Treated as final transaction state
41	Card acceptor terminal identification	ans 8	Acquirer, configured in terminal	Yes	Duplicated across sites
42	Card acceptor identification code	ans 15	Acquirer	Yes	Auth and clearing disagree
49	Currency code, transaction	n 3	Terminal configuration	Yes	Alphabetic code; default 840

DE	Name	Format	Written by	Echoed unchanged	Principal failure
55	Integrated circuit card data	ans…999, TLV	Card and terminal; issuer in response	Yes, byte for byte	Truncation; encoding; flat parsing

What this table leaves out

Fifteen fields is enough to move money and nowhere near enough to run an acquiring business. DE35 and DE45 carry track 2 and track 1 data. DE25 carries the point of service condition code, which says whether this is a normal sale, a recurring payment, a mail order or a repeat. DE32 and DE33 identify the acquiring and forwarding institutions and are part of the matching key. DE48 is additional data, private, and is where every scheme has quietly parked the things ISO 8583 has no room for. DE52 carries the enciphered PIN block, formatted per ISO 9564-1. DE54 carries additional amounts. DE61 and DE62 are private fields where Mastercard puts point of service data and transaction-life-cycle identifiers. DE64 and DE128 carry message authentication codes. DE90 carries the original data elements that let a reversal find what it is reversing.

Each has the same three questions attached to it: what it holds, who writes it, and what breaks when it is wrong. The habit is worth more than the list, because the list changes between schemes and the habit does not.

The next chapter takes a single authorisation and walks it end to end, byte by byte, with every field in this chapter populated and every hop accounted for. It is the same purchase, the same bookshop and the same £42.50, and by the end of it there will be nothing left in the message you cannot name.

26.98 Common wrong ideas

Wrong: every desk copies the boxes it did not write, so the message the issuer receives is the message the terminal sent. **Right:** acquirers renumber DE11, may replace DE2 with a network token, translate processing codes and add DE5 and DE50, so debugging across an acquirer boundary means comparing two different messages that describe one purchase.

Wrong: the field number determines the meaning. **Right:** ISO 8583 fixes the number, the name and the format but not the vocabulary inside; the complete question is which field, on which scheme, in which edition, in which direction.

Wrong: the issuer writes DE38 and DE39 and copies the rest. **Right:** the response can also carry a reduced amount for a partial approval, balances in DE54, issuer authentication data and scripts in DE55, and private data in DE48 that changes what the terminal must display.

Wrong: DE11 identifies a transaction. **Right:** the STAN is unique only within one originator within a processing day and wraps at 999999; it is one part of a matching key that conventionally also uses DE7, DE32 and DE37.

Wrong: a retransmission should carry a fresh STAN and a fresh retrieval reference number. **Right:** a repeat carries the same values with the MTI's origin digit changed from 0 to 1; regenerating them converts a timeout into a genuine double charge.

Wrong: amounts have two decimal places. **Right:** the exponent comes from the ISO 4217 entry for the currency in DE49 — none for the yen, three for the Kuwaiti dinar — and hard-coding two corrupts those amounts by orders of magnitude.

Wrong: `07` and `91` in DE22 are much the same, because the customer tapped the card either way. **Right:** `07` is a contactless chip transaction carrying a cryptogram in DE55 and `91` is a contactless read that produced stripe-equivalent data, which is a different security posture entirely.

Wrong: DE38 can be filled in locally when the issuer did not return one. **Right:** an invented authorisation code matches nothing at the issuer, and the clearing record carrying it is a candidate for a chargeback on the grounds of no valid authorisation.

Wrong: DE43 is cosmetic free text. **Right:** it is the raw material for the statement descriptor, and a descriptor reading as an unrecognisable holding company generates disputes at a multiple of the rate of one reading as the shop's trading name and town.

Wrong: DE55 can be parsed by position like the rest of the message. **Right:** it is BER-TLV under ISO/IEC 7816-6, where tags may legitimately be re-ordered and templates nest, so a flat positional parser will read an issuer script as a single opaque blob.

26.99 Chapter summary in 20 lines

1. A message that will not parse is a cheap problem; the expensive failures are the ones where every field parses perfectly and one value is simply the wrong kind of true.
2. There are 128 numbered data elements, a working card authorisation is built from about twenty of them, and about fifteen carry the weight.
3. The plain rule is that each desk along the chain writes its own boxes and copies the rest, with the till writing the facts of the sale, the acquirer the facts of the journey, and the issuer exactly two boxes.
4. That rule is the most useful lie in the chapter, because acquirers renumber the STAN, tokenise the account number, translate processing codes and add currency fields the terminal never sent.
5. A field definition is a character set and a length rule, where a bare number means fixed, two dots mean a two-digit length prefix and three dots mean a three-digit one.
6. Fixed numeric fields are right-justified and zero-filled while fixed alphanumeric fields are left-justified and space-filled, and that asymmetry causes real matching defects.
7. DE2 carries the account number, is frequently repopulated by tokenisation or point-to-point decryption, and is quite often absent altogether when full track 2 data travels in DE35 instead.
8. DE3 is three pairs — transaction type, account from, account to — and the transaction types diverge between networks publicly and expensively.
9. DE4 is twelve unsigned digits of minor units with no decimal separator, and the decimal position is implied by the ISO 4217 exponent of the currency named in DE49.
10. DE7, DE12 and DE13 are three different clocks — the sender's transmission time, the terminal's local time and the terminal's local date — and DE15, the settlement date, is a fourth date that determines when money moves.
11. The 1987-edition date fields carry no year, so a purchase at 23:58 local on 31 December transmitted at 00:02 UTC can be reconstructed 364 days wrong.
12. DE11 is a correlation key unique only within one originator for one processing day, and treating it as a transaction identifier is a design error that surfaces at scale.
13. DE22 states how the card was read, and because liability rules and interchange hang on it, one wrong digit is a commercial event rather than a technical one.
14. DE37 has the longest life in the message, running from authorisation into clearing, settlement and any dispute, and it must be echoed exactly, leading zeros and trailing spaces intact.
15. DE38 is the issuer's own handle on the approval it granted, it is alphanumeric so a numeric column will corrupt it, and it must never be invented downstream.

16. DE39 is two characters in the 1987 edition and three in 1993, the two vocabularies coexist, and the mapping between them is lossy in both directions.
17. DE41, DE42 and DE43 identify the terminal, the merchant and the statement descriptor, and forty characters of free text in DE43 drive a measurable share of disputes.
18. DE49 fixes the currency and by implication the decimal position, and a configuration template defaulting to `840` produces a valid message denominated in a currency the shop does not trade in.
19. DE55 is a format inside a format, a sequence of EMV tag-length-value objects whose amount, currency and country are the values the card actually signed, so a disagreement with DE4, DE49 or the country code tells the issuer that something altered the transaction.
20. Everything else in an authorisation is an unauthenticated assertion by whoever sent it, which is why card payments are governed by liability rules and scheme membership rather than by proof.

Chapter sources: ISO 8583:1987 and ISO 8583:1993 data element and response code tables as consolidated in the Wikipedia ISO 8583 article and cross-checked against ISO catalogue entries for ISO 8583:1993 and ISO 8583-1:2003, including clause 6.4.9 and Annex A.16 on the point of service data code; ISO 8583:2023 on the ICC data element and ISO/IEC 7816-6; SoFi Tech Solutions (Galileo) DE003 and DE022 code references, stated to be derived from the Mastercard Customer Interface Specification of 6 February 2024 and VisaNet Authorization-Only Online Messages of 15 October 2023; Mastercard IPM Clearing Formats on the Julian processing date in DE37; American Express Global Credit Authorization Guide, ISO 8583:1993 Version 1, October 2019, for the twelve positions of the point of service data code; Worldpay ISO 8583 Reference Guide v2.50 for acquirer-interface field attributes; EMV Integrated Circuit Card Specifications Book 3 tag definitions as tabulated by EMV Lab and EFTlab; ISO/IEC 7812-1, ISO/IEC 7813, ISO 4217 and ISO 9564-1.

CHAPTER 5

A Transaction, Narrated

27.0 What this chapter gives you

1. You will be able to narrate a contactless card authorisation from card presentation to receipt, naming every hop and what each party does in it.
2. You will be able to say where the 1.8 seconds actually goes, and why the printer is a larger component than the network, the acquirer, the scheme and the issuer put together.
3. You will be able to explain why the beep at 320 milliseconds is not an approval, and why nothing has left the shop when it sounds.
4. You will be able to say why the issuer's response cryptogram in tag `91` is discarded on a contactless tap and delivered to the card on a contact chip transaction, and what that costs in security and in time.
5. You will be able to expand a primary bitmap such as `723C648108E08201` by hand and list the twenty elements it declares.
6. You will be able to name which fields the acquirer adds, which it rewrites and which it passes through unchanged, and why the ones it passes through are the ones every downstream reconciliation depends on.
7. You will be able to describe how an issuer validates a cryptogram — master key, card key, session key, recomputation — and state the three things a match proves that nothing else in the message can.
8. You will be able to explain why an approval reduces an available balance and touches no ledger anywhere, and why no money has moved at 1,800 milliseconds.
9. You will be able to say what a terminal must do when no response arrives, and why retrying with a fresh trace number is the wrong answer.
10. You will be able to reason about the tail rather than the median — mobile data, cold DNS, PIN entry, stand-in processing — because that is where merchant contracts and scheme rules are written.

Everything in this volume so far has been apparatus. The four-party model gave you the cast. ISO 8583 gave you the envelope. The bitmap gave you the index. The fields gave you the contents. None of it has yet moved.

This chapter moves it. One purchase, one card, one terminal, one acquirer, one scheme, one issuer, and a stopwatch running from the moment the card enters the reader's field to the moment the receipt is torn off. Every message is written out. Every field is named and populated. Every hop is timed, and at the end the timings are added up and compared against the number everyone quotes and nobody audits, which is that a card payment takes about two seconds.

The purchase is the same one used in the previous two chapters, so that nothing has to be relearned: a book costing £42.50, bought at a small bookshop in London on Tuesday 17 March 2026, at eight minutes past half past two in the afternoon, paid for by tapping a contactless Visa debit card on a countertop terminal. The card number, `4111111111111881`, is from the test range every payments engineer recognises, and it is used here deliberately so that no real account appears in print.

Two warnings. The millisecond figures are a representative budget for a well-built United Kingdom deployment on a fixed broadband line, not a measurement of a particular system; the shape of the budget is the point, and the shape is stable even when the numbers move. And several values in the message are marked illustrative, because inventing an issuer's proprietary bytes and presenting them as fact is exactly what this book refuses to do.

The plain version

Imagine a shopkeeper who cannot accept your promise to pay, only your bank's.

You want a book. The shopkeeper wants £42.50. He does not know you, and he cannot look inside your bank account. So before he will let you leave with the book, he wants a sentence from your bank that says: *we have seen this, we have put £42.50 aside, and we will pay the shop*. Getting that one sentence, from a bank that might be four hundred miles away, is the entire event.

The trouble is that the shopkeeper cannot telephone your bank. There are thousands of banks and millions of shops, and nobody could keep that many phone numbers. So the question travels along a fixed chain of four parties, each of whom only ever talks to the party next to them.

The **card** is the first. It is not a piece of plastic with a number on it; it is a tiny computer that can perform a calculation nobody else can perform, because it holds a secret only it and your bank know.

The **till** — properly, the card terminal — is the second. It knows the price, the date, the shop, and how the card was presented. It cannot decide anything.

The **shop's bank**, the acquirer, is the third. It has an account for the shopkeeper, a direct line into the card networks, and a filing cabinet in which this transaction will live for years.

The **card company** in the middle, Visa in this case, is the fourth. It is a switchboard. Its job is to look at the front of the card number, work out which bank issued it, and pass the question on.

And **your bank**, the issuer, is the fifth and the only one that can say yes.

The question goes card, till, shop's bank, card company, your bank. The answer comes back your bank, card company, shop's bank, till. Nine steps in all, and it is over before you have finished putting your card away.

Where the time actually goes

Here is the same purchase with a stopwatch on it. Start the clock the instant the card comes within range of the reader.

What is happening	Time it takes	Clock reads
The card and the till talk to each other	0.32 s	0.32 s
The till writes out the question and sends it	0.08 s	0.40 s

What is happening	Time it takes	Clock reads
The shop's bank reads it, stamps it, forwards it	0.07 s	0.47 s
The card company works out which bank, forwards it	0.06 s	0.53 s
Your bank checks the card is genuine and the money is there	0.25 s	0.78 s
The answer travels all the way back to the till	0.12 s	0.90 s
The till reads the answer and puts APPROVED on the screen	0.03 s	0.93 s
The till prints the receipt	0.87 s	1.80 s

Read that table twice, because it says something nobody expects.

The part everyone imagines is slow — the message flying across the country to your bank and back — is the smallest number in it. All four outward journeys and all four return journeys, added together, come to about a tenth of a second. The computers at the other end are not slow either; your bank's whole decision takes a quarter of a second.

The two big numbers are the ones nobody thinks about. Nearly a fifth of the time is the card and the till talking to each other over a few centimetres of air before anything leaves the shop at all. And almost half of it is the printer.

That is worth saying plainly. In a modern card payment, the single slowest component is a strip of paper being pushed past a hot line of dots.

Three things that are not what they look like

The beep is not the approval. The terminal beeps and shows a green light at about 0.32 seconds, when the card has finished its part and can be taken away. At that moment nothing has left the shop. No bank has seen anything. The beep means "I have got what I need from your card", and people read it as "you have paid", which is why customers walk off and why terminals then have to display a decline to somebody's back.

Your bank is not looking up your balance and subtracting. It does two quite different things. It checks that the card is genuine, by redoing the calculation the chip performed and seeing whether it gets the same answer. And it checks the money is available, by looking at your balance and at everything else it has already promised out of it today. Then it *earmarks* £42.50.

No money has moved. At the end of all 1.8 seconds, the shopkeeper has a promise and you have a hold on your account. The £42.50 has not left your bank and it has not arrived at the shopkeeper's. That happens later, in a batch, overnight, and it is the subject of a later chapter. All this chapter's machinery exists to produce one six-character code and a two-character yes.

Where the plain version stops being true

There is no such thing as "the message". The plain version speaks of a question travelling along a chain, as though one document were being carried from hand to hand. Nothing of the sort happens. The terminal produces a message in one format; the acquirer parses it, discards some of it, adds fields the terminal never sent, rewrites fields the terminal did send, and produces a different message in a different format; the scheme does the same again on the way to the issuer. In the United Kingdom

this is unusually visible, because the terminal-to-acquirer link has historically not used ISO 8583 at all. It uses Standard 70, the Card Acceptor to Acquirer Interface Standards maintained on the industry's behalf by UK Payments Administration and its predecessors, whose Book 2 covers real-time systems and whose Book 3 covers post-event systems. A British terminal typically speaks Standard 70 to its acquirer, and the acquirer speaks ISO 8583 to the scheme. Two formats, two vocabularies, one purchase. When you debug across that boundary you are comparing two different documents, and the correct question is never "what did the message say" but "which message, at which hop".

The card is gone before the answer exists. In the plain version the chain is symmetrical: the question goes up and the answer comes back down the same path. For a contact chip transaction that is nearly true, because the card stays in the slot and receives the issuer's reply. For a contactless tap it is false in a way that matters. The card finishes its work at around 320 milliseconds and is back in your pocket by 500. The issuer's reply, which contains a cryptogram the card could in principle have verified, arrives at the terminal at around 900 milliseconds with nothing to deliver it to. The terminal discards it. This is not a defect; contactless was designed this way on purpose, trading one protection for the speed that makes tapping tolerable at a ticket barrier. But it means the security model of a tap and the security model of an inserted chip are genuinely different, and the difference is invisible to everyone in the shop.

These are not constants; they are distributions with a long tail. Every number in the plain version's table is roughly a median. The engineering discipline of payments is not about medians. An acquirer's contract with a merchant, and a scheme's rules for an issuer, are written about the far end of the distribution: how often does this exceed one second, exceed three, time out altogether. The interesting behaviour is all in that tail, and it is where the protocol's timeouts live. Mastercard's published documentation for one of its processing products states that the issuer must respond within four seconds before that processor stands in on the issuer's behalf, and that the processor in turn has a maximum of seven seconds to respond to Mastercard before Mastercard stands in. Those numbers exist because the tail exists. A system built to the median will meet them comfortably in testing and breach them on the first Saturday before Christmas.

Nothing in the narrative is the transaction. The receipt feels like the end. It is the end of the authorisation and the beginning of everything else. The clearing record has yet to be created, the interchange has yet to be assessed, the funds have yet to settle, the hold has yet to be released or converted, and the transaction remains disputable for months. What has been narrated here is the fastest and most reliable part of the life of a payment, and it is the part on which the industry's public reputation rests, which is precisely why the slow and expensive parts that follow are so poorly understood.

The technical version

The setting

A countertop terminal in a bookshop in London. It is EMV Level 1 and Level 2 approved, contact and contactless, PCI PTS approved, with a thermal printer and a wired Ethernet connection to a business broadband line. It is configured for a single United Kingdom acquirer.

The card is a consumer Visa debit card, issued in the United Kingdom, dual interface, supporting Visa's contactless application. Its application identifier is `A0000000031010`.

The date is Tuesday 17 March 2026, day 076 of the year. British Summer Time does not begin until Sunday 29 March 2026, so on this date the United Kingdom is on Greenwich Mean Time and local

time equals Coordinated Universal Time exactly. This is a convenience that disappears twelve days later, and the disappearance breaks a great many reconciliation jobs each spring.

The card is presented at 14:32:04.800 local. That instant is T+0 for everything that follows.

The transaction is £42.50. The relevant United Kingdom contactless limits, as of writing in 2026, are a single-transaction ceiling of £100, in force from 15 October 2021 until 18 March 2026, after which the Financial Conduct Authority permitted firms to remove the mandatory cap; and a cumulative threshold of £300, or five consecutive contactless transactions, since the last application of strong customer authentication, the cumulative figure having been raised from £130 by the FCA in policy statement PS21/2. On 17 March 2026 the £100 ceiling still applies, and £42.50 is inside both, so no cardholder verification will be requested.

Phase one, T+0 to T+320 ms: the card in the field

The reader has been polling continuously. The card's antenna couples to the reader's field at 13.56 MHz under ISO/IEC 14443, draws its power from that field, and answers.

Elapsed	Step
0 to 35 ms	Field detection, ISO/IEC 14443-3 anticollision, answer to select
35 to 65 ms	SELECT of the proximity payment system environment, `2PAY.SYS.DDF01`
65 to 95 ms	SELECT of the application identifier `A0000000031010`
95 to 255 ms	GET PROCESSING OPTIONS, carrying the PDOL; card computes and returns the cryptogram
255 to 300 ms	READ RECORD for the remaining card data
300 to 320 ms	Kernel completes, terminal beeps and shows the green light

Three details of that sequence do real work.

The terminal does not ask for the payment application by name. It selects the proximity payment system environment, `2PAY.SYS.DDF01`, reads back the list of applications the card supports, and picks one by its own configured priority. This is why a card carrying two applications can behave differently at two terminals that are otherwise identical.

The GET PROCESSING OPTIONS command is where the transaction is actually committed. The card publishes a processing options data object list, the PDOL, naming the values it wants from the terminal. For Visa's contactless kernel this list includes the terminal transaction qualifiers in tag `9F66`, the authorised amount in `9F02`, the other amount in `9F03`, the transaction currency in `5F2A`, the terminal country in `9F1A`, the unpredictable number in `9F37` and the transaction type in `9C`. The terminal concatenates those values in the order the card asked for them and sends them. The card signs over them. That is the whole basis of the cryptogram's meaning: it attests to *these values*, in this order, at this moment.

And in Visa's contactless EMV mode the cryptogram comes back in the response to GET PROCESSING OPTIONS. There is no separate GENERATE AC exchange, and therefore no second exchange in which the issuer's reply could be handed to the card. This is the single design decision that makes contactless fast and that makes the issuer's response cryptogram undeliverable.

At 320 milliseconds the terminal beeps and the customer takes the card away. The terminal now holds an authorisation request cryptogram, tag `9F26`, eight bytes, computed by the chip using a key derived from a master key held by the issuer, together with the transaction counter in `9F36` that makes this cryptogram different from every other one this card has produced.

Phase two, T+320 to T+400 ms: the terminal builds the request

The terminal now performs terminal action analysis. It examines the terminal verification results in tag `95`, five bytes of flags, and the cryptogram information data in tag `9F27`, one byte, whose value indicates an authorisation request cryptogram. It applies its floor limit, which for a United Kingdom contactless acceptance point is effectively zero, and concludes what it already knew: this transaction must go online.

It stamps the local transaction time and date. Its clock reads 14:32:05.120, so data element 12 becomes `143205` and element 13 becomes `0317`. It assigns its own system trace audit number, `000982`, from a counter persisted to non-volatile storage. It composes the message, applies whatever point-to-point encryption its acquirer requires, and hands it to the network stack.

At 375 milliseconds the request goes out over an already-established TLS session. A terminal that keeps a persistent, authenticated connection to its acquirer pays the handshake cost once, at sign-on. A terminal that opens a fresh connection per transaction pays it every time: a TLS 1.3 handshake costs one round trip before application data can flow, TLS 1.2 costs two, and both cost certificate validation on a processor that is not fast.

The one-way transit to the acquirer's front end, over business fibre to a London data centre, takes about 25 milliseconds including the shop's own router. Arrival at 400 milliseconds.

The message as it leaves the terminal

Presented here in ISO 8583 terms for comparability with what follows, although a British terminal would more usually present it in Standard 70 form.

MTI `0100`, authorisation request: version 0 for the 1987 edition, class 1 for authorisation, function 0 for request, origin 0 for acquirer.

Primary bitmap `723C448000E08200`, declaring sixteen elements present.

DE	Value	Note
2	`16 4111111111111881`	LLVAR, two-digit length then the PAN
3	`000000`	Purchase, from default account, to default account
4	`000000004250`	4,250 minor units
7	`0317143205`	MMDDhhmmss, UTC, stamped by the terminal
11	`000982`	The terminal's own trace number
12	`143205`	Local transaction time
13	`0317`	Local transaction date
14	`2809`	Expiry, YYMM
18	`5942`	Merchant category code, book stores, per ISO 18245
22	`071`	Contactless chip, terminal has PIN entry capability
25	`00`	Point of service condition code, normal transaction
41	`TERM0042`	Eight characters, left justified, space filled
42	`KEDBYTEBOOKS01`	Fifteen characters, one trailing space
43	`KEDBYTE BOOKS  LONDON  GB`	Forty characters
49	`826`	ISO 4217 numeric for sterling

DE	Value	Note
55	160 bytes of BER-TLV	See below

Data element 43 deserves a note. Its forty characters are subdivided into name, town and country, but the subdivision is scheme-defined and the schemes do not agree on the boundaries; the value above is padded on the convention of twenty-five characters of name, thirteen of town and two of country code.

Two coincidences in the message are not coincidences. Data element 49 carries `826` for sterling and data element 19, which the acquirer will add shortly, carries `826` for the United Kingdom. ISO 4217's numeric currency codes were assigned to match ISO 3166-1's numeric country codes wherever a country has its own currency, so the sterling code and the British country code are the same three digits by construction. Systems that validate one against the other by equality will work perfectly in Britain and fail in the eurozone.

Data element 55, in full

The acquirer interface used here encodes data element 55 as a three-character ASCII total length followed by binary tag-length-value data, a convention documented in Worldpay's Lync ISO 8583 message specification and typical of the industry. The tag-length-value structure is BER-TLV as defined for EMV, not the composite-dataset structure that ISO 8583:2023 defines for its own fields; the 2023 standard explicitly exempts the integrated-circuit-card element and defers to ISO/IEC 7816-6.

Twenty-four objects, 160 bytes:

Tag	Bytes of value	Content	Exact here?
`9F26`	8	Application cryptogram, the ARQC	Illustrative
`9F27`	1	Cryptogram information data, indicating an ARQC	Exact
`9F10`	7	Issuer application data	Illustrative, card profile dependent
`9F37`	4	Unpredictable number	Illustrative
`9F36`	2	Application transaction counter	Illustrative
`95`	5	Terminal verification results	Illustrative
`9A`	3	Transaction date, `260317`	Exact
`9C`	1	Transaction type, purchase	Exact
`9F02`	6	Amount authorised, `000000004250`	Exact
`9F03`	6	Amount other, `000000000000`	Exact
`5F2A`	2	Transaction currency code, `0826`	Exact
`82`	2	Application interchange profile	Illustrative
`9F1A`	2	Terminal country code, `0826`	Exact

Tag	Bytes of value	Content	Exact here?
9F33	3	Terminal capabilities	Illustrative
9F34	3	Cardholder verification method results	Illustrative
9F35	1	Terminal type	Illustrative
84	7	Dedicated file name, A0000000031010	Exact
9F09	2	Application version number	Illustrative
9F41	2	Transaction sequence counter	Illustrative
5F34	1	Application PAN sequence number	Illustrative
9F6C	2	Card transaction qualifiers	Illustrative
9F66	4	Terminal transaction qualifiers	Illustrative
9B	2	Transaction status information	Illustrative
57	19	Track 2 equivalent data	Illustrative

Note tag 57. On a chip transaction the track 2 equivalent data lives inside data element 55, and the acquirer interface used here specifies that where tag 57 is present it is the source of track data. That is why the message carries no data element 35: duplicating it would create two sources of truth for the same account number.

Note also 9F02, 5F2A and 9F1A. The amount, the currency and the country each appear twice in this message: once in the ISO 8583 elements 4, 49 and 19, and once inside the cryptogram's input. Only the second copy is protected. If those two copies disagree, the issuer knows that something between the chip and itself altered the transaction, and it will decline. This is the only integrity guarantee in the entire message that does not depend on trusting the sender.

The same acquirer specification states that data element 55 is required on 0100, 0200, 0220 and 0400 messages and on stand-in messages, and must not be sent to the network for manual or fallback transactions. That last clause is the certification failure named in the previous chapter, in its formal wording.

Total message length as it leaves the terminal, in the ASCII-and-hexadecimal representation used here: 484 characters, of which 323 are data element 55.

Phase three, T+400 to T+470 ms: the acquirer

The acquirer's front end does more in seventy milliseconds than any other party in the chain.

It terminates the TLS session and parses the message, defensively: length prefixes are the standard attack surface of this protocol and a hostile or broken length is the first thing a front end must survive. It performs duplicate detection against the terminal's trace number and terminal identifier over a short window, because the commonest cause of a double charge is not fraud but a terminal that retried.

It decrypts. If the account data arrived under a point-to-point encryption scheme, the ciphertext goes to a hardware security module — a tamper-responsive device holding keys that cannot be read out of it in the clear — and comes back as a primary account number inside the acquirer's secure zone. That round trip is a few milliseconds and it is the moment at which the acquirer's PCI DSS scope is defined.

Had this been a PIN transaction, the same module would also have translated the PIN block from the key shared with the terminal into the key shared with the scheme, without the PIN ever existing in plaintext outside a tamper boundary.

It applies its own risk rules: velocity by terminal, by card, by merchant, and amount thresholds. A small proportion of traffic dies here and never reaches a scheme.

Then it writes its fields.

- Data element 7 is restamped with the acquirer's own transmission time.
- Data element 11 is replaced. The terminal's `000982` is meaningful only within that terminal; the acquirer substitutes `004417` from a counter unique within the acquirer for the processing day.
- Data element 19 is set to `826`, the acquiring institution's country.
- Data element 32, the acquiring institution identification code, is populated with the acquirer's scheme-assigned identifier. This is `060000` in the worked example and is illustrative.
- Data element 37 is created: `607614430041`, decomposing as year digit 6, day 076, hour 14 and a sequence. This value will now outlive everything else in the message.
- Data element 15, the settlement date, may be set here, determining which business day this transaction belongs to regardless of what the clocks say.
- Data element 64, an eight-byte message authentication code, is computed over the assembled message under a key shared with the scheme.

At 455 milliseconds the acquirer's routing logic reads the leading digits of the account number, identifies a Visa product, and hands the message to its scheme endpoint. Twenty digits of routing decision, made on a table that is refreshed daily and that every acquirer in the world maintains separately.

Fifteen milliseconds of transit. Arrival at the scheme, 470 milliseconds.

The message as it leaves the acquirer

MTI `0100`. Primary bitmap `723C648108E08201`.

That hexadecimal string is worth expanding by hand, because it is the whole argument of the bitmap chapter reduced to eight bytes.

Byte	Hex	Bits	Elements present
1	72	01110010	2, 3, 4, 7
2	3C	00111100	11, 12, 13, 14
3	64	01100100	18, 19, 22
4	81	10000001	25, 32
5	08	00001000	37
6	E0	11100000	41, 42, 43
7	82	10000010	49, 55
8	01	00000001	64

Twenty elements declared in sixty-four bits. Bit 1 is zero, so there is no secondary bitmap and nothing above element 64 is present.

The message now runs to 523 characters. Compared with what the terminal sent, four elements have been added (19, 32, 37, 64) and two have been rewritten (7, 11). Sixteen have been passed through unchanged, and it is on those sixteen that every downstream reconciliation depends.

Phase four, T+470 to T+525 ms: the scheme routes it

The scheme is a switch, and its work is narrow, fast and unglamorous.

It validates the message authentication code in data element 64 against the key it shares with this acquirer. It reads the issuer identification number from the leading digits of data element 2 — under ISO/IEC 7812-1 as revised in 2017 these are eight digits, though six-digit ranges remain in wide use and both must be supported — and looks up the destination in a routing table.

If the account number in data element 2 is a network token rather than a funding account number, the scheme's token vault resolves it here, and the payment account reference, a value that stays constant across every token issued against the same underlying account, travels alongside so that the merchant and the issuer can still tell that two different tokens are the same customer. In the acquirer interface used above this reference occupies data element 56.

The scheme scores the transaction against its own risk models, starts a response timer, and records enough to build a clearing record later and to answer on the issuer's behalf if the issuer does not reply. Then it sends. VisaNet, by Visa's own published figures, is engineered to handle more than 65,000 transaction messages per second; this is one of them, and it occupies the switch for perhaps thirty-five milliseconds.

Twenty milliseconds of transit to the issuer's processor. Arrival at 525 milliseconds.

Phase five, T+525 to T+775 ms: the issuer decides

This is the largest single block of computation in the transaction, and it is a quarter of a second.

Elapsed	Step
525 to 540 ms	Parse, validate, check for a duplicate against DE11, DE37 and DE32
540 to 570 ms	Validate the cryptogram in a hardware security module
570 to 615 ms	Read the account, the balance and the outstanding holds
615 to 700 ms	Score the transaction against the fraud model
700 to 740 ms	Decide, place the hold, memo-post, generate DE38 and DE39
740 to 760 ms	Generate the response cryptogram in the security module
760 to 775 ms	Build and transmit the `0110`

The cryptogram validation is the part worth understanding in detail, because it is the only step in the whole chain that proves anything.

The issuer holds a master key for the card portfolio inside a hardware security module. From that master key, together with the account number and the application PAN sequence number in tag `5F34`, the module derives the unique key belonging to this individual card. From that card key, together with the application transaction counter in tag `9F36`, it derives a session key that is used for this transaction and no other. It then recomputes the cryptogram over the data the card signed — `9F02`, `9F03`, `5F2A`, `9F1A`, `95`, `9A`, `9C`, `9F37`, `82`, `9F36` and `9F10` — and compares its answer with the eight bytes in `9F26`.

If those eight bytes match, the issuer knows three things it cannot know any other way: that a genuine chip carrying its key was present; that the chip saw this amount and this currency; and that this is not

a replay, because the counter has advanced. If they do not match, everything else in the message is unsupported assertion.

The balance check is not a subtraction from a single number. The issuer holds a ledger balance, which is what has actually posted, and computes an available balance by subtracting authorisations already granted and not yet cleared. The £42.50 becomes another such deduction: an entry in a shadow-balance table with an expiry, not a movement in the ledger.

The fraud model is where the variance lives. A real-time scoring service consulting device, merchant, geography and behavioural features is the likeliest reason an issuer's median response time is 250 milliseconds rather than 90, and its failure mode is a timeout rather than an error.

The decision is yes. The issuer generates the authorisation identification response, six alphanumeric characters, `A4T29B`, and sets the response code to `00`. It generates an authorisation response cryptogram in the security module and packages it, with a response code the card would understand, into tag `91` inside data element 55.

The response

MTI `0110`, authorisation response: same version, same class, function 1 for request response, origin 0 because it retains the acquirer origin of the request.

Primary bitmap `723860010EC08201`.

Byte	Hex	Bits	Elements present
1	`72`	01110010	2, 3, 4, 7
2	`38`	00111000	11, 12, 13
3	`60`	01100000	18, 19
4	`01`	00000001	32
5	`0E`	00001110	37, 38, 39
6	`C0`	11000000	41, 42
7	`82`	10000010	49, 55
8	`01`	00000001	64

Eighteen elements, 186 characters. Of those eighteen, exactly two are new:

DE	Value	Meaning
38	`A4T29B`	Authorisation identification response, the code printed on the receipt
39	`00`	Approved or completed successfully

Data element 55 in the response contains a single object: tag `91`, issuer authentication data, eight or ten bytes depending on which response cryptogram method the card profile uses. Everything else is echo.

Data element 4 is echoed at `000000004250`. Had the issuer approved a lesser sum, this field would have come back smaller, and the terminal would have been required to notice, tell the operator, and collect the difference or reverse the whole transaction. A terminal that reads only data element 39 and ignores data element 4 will approve a partial authorisation as though it were full and leave the merchant short.

Phase six, T+775 to T+895 ms: back down the chain

Elapsed	Step
775 to 795 ms	Issuer to scheme, transit
795 to 815 ms	Scheme stops its timer, logs, seeds the clearing record, revalidates the MAC
815 to 830 ms	Scheme to acquirer, transit
830 to 870 ms	Acquirer records the authorisation, maps DE39 to its terminal vocabulary, rebuilds the response
870 to 895 ms	Acquirer to terminal, transit

The acquirer's forty milliseconds are not idle. It writes an authorisation record keyed on data element 37 that will have to be matched against a clearing record arriving days later, it reserves the merchant's exposure, and it translates its terminal-facing response vocabulary — a mapping that is lossy in both directions for every value except approval.

Phase seven, T+895 to T+1,800 ms: the terminal finishes

The terminal parses the response, confirms it matches the request it is waiting on, and updates its state machine to approved. At 935 milliseconds the screen reads APPROVED and the operator can start bagging the book.

It also discards tag 91. The card left the field six hundred milliseconds ago, and there is no second exchange in a contactless transaction to deliver the issuer's cryptogram to. Had this been a contact chip transaction the terminal would now issue a second GENERATE AC command carrying that data, and the card would verify the issuer's cryptogram and return either a transaction certificate or an application authentication cryptogram — retaining the right to decline a transaction the issuer has just approved. That exchange costs a further two to four hundred milliseconds and requires the card to still be in the slot, which is the entire reason chip-and-PIN feels slower than tapping.

Then it prints. A thermal printer laying down a merchant copy of roughly 120 millimetres at a realistic line speed takes about 660 milliseconds, and cutting, clearing and prompting for the customer copy takes another 200. Receipt out at 1,800 milliseconds.

The printed record carries data element 38 as the authorisation code, elements 12 and 13 as the date and time, element 41 as the terminal identifier, a masked form of element 2, and the entry mode derived from element 22. It does not carry element 37, which is the one number the acquirer will ask the merchant for when something goes wrong. That omission is a small, permanent, industry-wide own goal.

Where the two seconds actually goes

Component	Milliseconds	Share
Card and terminal, over the air	320	18%
Terminal software, both directions	95	5%
Network transit, all six one-way hops	120	7%
Acquirer processing, both directions	95	5%
Scheme processing, both directions	55	3%
Issuer decision	250	14%
Printer and paper	865	48%
Total	**1,800**	**100%**

Three conclusions follow, and all three are unwelcome to somebody.

The network is not the problem. Six one-way hops across two commercial boundaries, including a leg into a global switch, come to 120 milliseconds. That is under seven per cent, and it is the component with the least headroom, because most of it is the speed of light in glass.

The issuer is not the problem either, though it is the largest computational block. A quarter of a second to validate a cryptogram in a hardware security module, read an account, score it against a fraud model and commit a hold is not slow; it is the cost of doing the work honestly.

The problem, if there is one, is physical. The card conversation and the printer together are two thirds of the elapsed time, and neither is a computer being slow. One is a passive device drawing its power from a radio field. The other is paper. This is why the most effective latency work in retail payments over the past decade has been suppressing the customer receipt and printing the merchant copy asynchronously, which removes up to 865 milliseconds from the queue without touching a line of protocol code.

The tail

Everything above is a median. Here is the same transaction going wrong in the ordinary ways, and what each costs.

Condition	Added time
Terminal on a mobile data connection instead of fixed broadband	400 to 1,200 ms
Fresh TLS handshake per transaction rather than a persistent session	60 to 200 ms
DNS resolution with a cold or expired cache	20 to 500 ms
Contact chip rather than contactless, second GENERATE AC exchange	200 to 400 ms
Cardholder verification required, PIN entered	2,000 to 6,000 ms, human-limited
Issuer fraud model cold path or garbage collection pause	200 to 800 ms
Issuer times out and the scheme stands in	Up to the scheme's timer

The last row is the one that changes the shape of the outcome rather than the duration. Mastercard's published documentation for one of its processing products puts the issuer response window at four seconds, and the processor's own window at seven seconds, after which Mastercard authorises on the issuer's behalf. When that happens, the approval the merchant receives was granted by the scheme against limits the issuer configured in advance, the issuer learns about it afterwards by advice message, and the liability position is different from a normal approval in ways set out in the scheme's rules rather than in any technical specification.

Note also the cardholder verification row. The moment a human is asked to do something, every millisecond above becomes rounding error, and no amount of protocol optimisation touches it.

When the answer never comes

The narration above has a response arriving. Consider the case where it does not: the acquirer's reply is lost between 870 and 895 milliseconds, after the issuer has already placed a hold.

The terminal times out. It does not know whether the transaction was approved, declined, or never seen. What it must not do is simply retry with a fresh trace number, because the issuer would then see two unrelated requests and place two holds.

The correct behaviour is one of two things. Either it retransmits the identical message with the MTI's origin digit changed from 0 to 1, producing `0101`, with the same trace number and retrieval reference number, so the issuer recognises a repeat of something it may already have answered. Or it sends a reversal, MTI `0400`, quoting the original message's identifying values inside data element 90 — the original MTI, trace number, transmission date and time, and acquiring and forwarding institution identification codes — so the issuer can find the hold and release it.

Reversals are the least glamorous and most consequential messages in the protocol. A terminal that fails to send them leaves customers with money held against purchases that never happened, for as long as the issuer's hold expiry allows. Every complaint that begins "the payment failed but the money has gone" is a missing `0400`.

What has and has not happened

At 1,800 milliseconds the customer has a book, the operator has a receipt, and the following is true.

The issuer has reduced the cardholder's available balance by £42.50 and has not touched the ledger balance. The acquirer holds an authorisation record keyed on `607614430041` with an approval code of `A4T29B`. The scheme holds a routing and timing record. The terminal holds a batch entry awaiting the day's close.

No money has moved. No entry has been posted to any account at the Bank of England. The merchant will not be paid until the transaction is submitted for clearing, matched, assessed for interchange — at, as of writing, a United Kingdom domestic cap of 0.2 per cent for consumer debit where merchant, acquirer and issuer are all in the United Kingdom, which on £42.50 is 8.5 pence — and settled between the parties' banks. On this transaction the merchant service charge, the scheme fees and the acquirer's margin will all be assessed against a payment that took 1.8 seconds to authorise and will take days to complete.

The next chapter deals with the identifiers left behind — the trace number, the retrieval reference number, the acquirer reference number and the authorisation code — because those four values are the only handles anybody has on this transaction for the rest of its life, they have four different owners and four different lifetimes, and asking the wrong party for the wrong one is how a simple query becomes a three-week correspondence.

27.98 Common wrong ideas

Wrong: one message travels along the chain from the till to the issuer. **Right:** each hop parses, discards, rewrites and re-emits, and in Britain the terminal typically speaks Standard 70 to its acquirer while the acquirer speaks ISO 8583 to the scheme, so the correct question is never "what did the message say" but "which message, at which hop".

Wrong: the network is the slow part of a card payment. **Right:** all six one-way hops come to about 120 milliseconds, under seven per cent of the elapsed time, while the printer is 865 milliseconds and nearly half.

Wrong: the beep means the payment is approved. **Right:** at 320 milliseconds the card has merely finished its part and nothing has left the shop, which is why customers walk away before a decline can be displayed.

Wrong: a tap and an inserted chip have the same security model. **Right:** in Visa's contactless mode the cryptogram arrives in the response to GET PROCESSING OPTIONS, there is no second GENERATE AC exchange, and the issuer's reply reaches a terminal with nothing left to deliver it to.

Wrong: the till's trace number identifies the transaction all the way through. **Right:** the acquirer replaces DE11 with a value unique within itself and creates DE37, which is the number that outlives everything else in the message.

Wrong: DE39 `00` is the whole answer, so the terminal can print APPROVED. **Right:** the response can echo a smaller DE4 for a partial approval, and a terminal that reads only DE39 will treat it as a full authorisation and leave the merchant short.

Wrong: the amount, currency and country the issuer checks are elements 4, 49 and 19. **Right:** only the copies inside DE55 — tags `9F02`, `5F2A` and `9F1A` — are covered by the cryptogram, and a disagreement between the two copies is exactly what an altered transaction looks like.

Wrong: the approval means money has moved. **Right:** the issuer has written an entry with an expiry into a shadow-balance table and has not touched the ledger balance, and nothing posts until clearing and settlement days later.

Wrong: when the response times out the terminal should send the transaction again. **Right:** it must either retransmit the identical message as a `0101` with the same trace and retrieval reference numbers, or send a `0400` reversal quoting the original values in DE90, because a fresh trace number produces two holds.

Wrong: the median timings are the engineering target. **Right:** contracts and scheme rules are written about the tail, where Mastercard's published four-second issuer window and seven-second processor window live, and a system built to the median will meet them in testing and breach them on the first Saturday before Christmas.

27.99 Chapter summary in 20 lines

1. One purchase — a £42.50 book tapped on a countertop terminal in London on 17 March 2026 — takes 1,800 milliseconds from card presentation to receipt.
2. The card is a computer holding a key only it and the issuer know, the terminal knows the context and can decide nothing, the acquirer owns the merchant relationship, the scheme is a switchboard, and only the issuer can say yes.
3. The first 320 milliseconds are the card and the reader talking over a few centimetres of air: anti-collision, selection of the proximity payment system environment, application selection, and GET PROCESSING OPTIONS.
4. The card publishes a processing options data object list naming the values it wants, the terminal supplies them in that order, and the cryptogram in tag `9F26` attests to exactly those values at that moment.
5. In Visa's contactless mode the cryptogram comes back in the GET PROCESSING OPTIONS response, so there is no second exchange and no way to deliver the issuer's reply to the card.
6. The terminal performs terminal action analysis, stamps its local time and date, assigns its own trace number, and sends the request at 375 milliseconds over an already-established TLS session.
7. Its message declares sixteen elements and runs to 484 characters, of which 323 are data element 55.
8. The acquirer's seventy milliseconds are the busiest in the chain: parse defensively, detect duplicates, decrypt in a hardware security module, apply risk rules, then write its own fields.
9. It adds elements 19, 32, 37 and 64, rewrites 7 and 11, and passes sixteen elements through unchanged, and it is on those sixteen that every downstream reconciliation depends.
10. The scheme validates the message authentication code, reads the issuer identification number from the leading digits of the account number, resolves a token if there is one, scores the transaction, starts a response timer and forwards it.

11. The issuer's quarter of a second is the largest computational block: parse, validate the cryptogram, read the account and its holds, score for fraud, decide, place the hold and build the `0110`.
12. Cryptogram validation derives a card key from a portfolio master key and a session key from the transaction counter, recomputes the cryptogram and compares it, proving that a genuine chip was present, that it saw this amount and currency, and that this is not a replay.
13. The balance check subtracts authorisations already granted from a ledger balance, and the £42.50 becomes another such deduction with an expiry rather than a movement in any ledger.
14. The response carries exactly two new fields, the authorisation code `A4T29B` in element 38 and the response code `00` in element 39, plus tag `91` inside element 55 that a contactless terminal will discard.
15. Adding the components up: the card and terminal 320 milliseconds, all network transit 120, the acquirer 95, the scheme 55, the issuer 250, terminal software 95, and the printer 865.
16. The two slowest components are not computers being slow but a passive device drawing power from a radio field and a strip of paper, which is why suppressing the customer receipt is the most effective latency work of the past decade.
17. The tail is where the engineering is: mobile data, per-transaction TLS handshakes, cold DNS, the contact chip's second GENERATE AC, and above all a human being asked to enter a PIN.
18. When the issuer does not answer within its window the scheme stands in, and the approval the merchant receives was granted against limits the issuer configured in advance, with a liability position set by scheme rules rather than by any technical specification.
19. When no response reaches the terminal the correct behaviour is a `0101` repeat carrying the same identifiers or a `0400` reversal quoting the original data elements in DE90, and every complaint that begins "the payment failed but the money has gone" is a missing `0400`.
20. At 1,800 milliseconds the customer has a book and the merchant has a promise, but no money has moved, no entry has been posted at the Bank of England, and the transaction remains disputable for months.

Chapter sources: ISO 8583:1987 data element and MTI tables as consolidated in the Wikipedia ISO 8583 article; Worldpay Lync ISO 8583 Message Specification, data element 55, for the DE55 subelement encoding scheme, the tag and length table including 9B, 9F66 and 9F6C, the treatment of tag 57 as the source of track data, and the message types requiring DE55; Visa Developer Request and Response Codes for point of service condition codes 00, 08 and 51; UK Payments Administration Standard 70, Books 2 and 3, as catalogued at iso8583.info; Mastercard Processing Debit API documentation for the four-second issuer and seven-second processor windows and stand-in processing; Visa's published figure of more than 65,000 transaction messages per second; FCA policy statement PS21/2 for the £300 cumulative contactless threshold, and reporting of the £100 limit in force from 15 October 2021 to 18 March 2026; Payment Systems Regulator on the UK Interchange Fee Regulation caps; ISO 18245 code 5942; EMVCo contactless entry point documentation and the arXiv systematisation "SoK: Security of EMV Contactless Payment Systems" for the PPSE, application selection and GET PROCESSING OPTIONS flows; ISO/IEC 14443, 7812-1, 7816-6, ISO 3166-1 and ISO 4217. All bitmap expansions, message lengths, timing totals and percentages were computed from the field sets stated in the text.

CHAPTER

6

The Number You Were Given

28.0 What this chapter gives you

1. You will be able to name the four identifiers a card payment leaves behind, say who created each one, where it appears and how long it lives.
2. You will be able to explain why searching on a system trace audit number alone returns every terminal's `004821` for every day in the retention window, and what the composite key should be instead.
3. You will be able to say why the approval code on a customer's receipt is corroboration and never a lookup key.
4. You will be able to tell a customer whether they were charged twice or are looking at one charge and one hold, and why refunding an uncleared authorisation leaves them net up.
5. You will be able to say which number to quote to your acquirer, which to the scheme, which to the issuer, and which never to ask a customer for.
6. You will be able to store each identifier at the width and type its specification demands, and say why twenty-three digits will not fit in a 64-bit integer.
7. You will be able to explain why an acquirer reference number exists if and only if the transaction cleared, and what that means for the commonest class of complaint.
8. You will be able to say why both major schemes had to invent identifiers of their own, and why Mastercard's has to be assembled from three fields.
9. You will be able to contrast a card payment's identifiers with a UETR, and say why a Swift payment can be tracked end to end and a card payment cannot.
10. You will be able to read a signature from a pair of records — sequential trace numbers seconds apart, two terminal identifiers, an incremental authorisation, a tip adjustment, a late reversal — and name the cause.

The plain version

Imagine a school trip. Forty children, one coach, one museum, one long day.

Before the coach leaves, the teacher walks down the aisle handing out numbers. "You're fourteen today." It is written on a sticker. It is used all day for one purpose only: counting heads. Every time the coach stops, the teacher calls the numbers and waits for forty voices. Number fourteen is not who you are. It is where you sit in the count. Next week, on the next trip, somebody else will be fourteen. On the coach going home, you might even be given a different number, because a different teacher is counting.

At the museum door you are given a wristband. It has a code printed on it, something like `M-3040-118`. It goes on at nine in the morning and stays on until you cut it off at home. It is yours for the whole

day, and no other child in the building has the same one. If you get lost, the wristband is what the staff read out over the radio.

Inside, at the gift shop, you buy a keyring. The person at the till presses a button, a machine thinks for a second, and a small slip prints with a code on it: `07B4C1`. That code means the shop's bank said yes. It does not say what you bought, or how much, or who you are. It only says: at this moment, a machine said yes to something.

Then, a week later, the school office does the accounts. Somebody sits down with a stack of receipts and writes each one into a ledger, giving each line its own long reference. That reference did not exist on the day. It was born a week later, in a different building, made by a different person, for a different purpose: settling up. But if you later ring the school and say the gift shop charged you twice, that ledger reference is the only number that gets you anywhere, because it is the one attached to actual money that actually moved.

Four numbers. The coach number, the wristband, the till slip code, the ledger reference. They are not four copies of the same thing. They have four different owners, four different lifetimes, and four different jobs, and if you ask the wrong person for the wrong one you get a shrug.

A card payment works exactly like this.

Say you buy lunch at a café in Leeds on Tuesday 3 March. The bill is £42.60. You tap your card. Between the tap and the receipt printing, four numbers come into existence, or will do shortly.

The first is the card machine's own counter. The terminal has been sitting there all morning ticking upward: 004819, 004820, and now yours, **004821**. It puts that number in the message it sends off, and when the answer comes back, that answer carries 004821 too. That is the entire point of it. It is how the terminal knows the reply that just arrived is the reply to *your* lunch, and not to the person behind you buying a coffee. It is the coach number. It counts up to 999999 and then rolls back round to the beginning and starts again. Yesterday's 004821 was somebody else's sandwich.

The second is a longer reference, twelve characters, something like **506201304821**. The café's bank makes this one. Unlike the counter, it is meant to last. It is the wristband. It goes onto the request, it comes back on the answer, and it stays attached to the record of your lunch when that record is filed away at the end of the day. If, six weeks later, somebody has to go and find the exact transaction in the café's bank's systems, this is the number they search on.

The third is the approval code: **07B4C1**. Six characters. Your own bank makes this one, and only makes it if the answer was yes. It appears on the paper receipt, which is why it is the number a customer is most likely to be holding when they ring you up. It is the till slip code from the gift shop. It proves that a yes happened. On its own it does not prove what the yes was *for*.

The fourth does not exist yet. At the end of the day the café's card machine closes its batch and sends the day's takings off to be paid. Somewhere in that process, a twenty-three digit reference is created for your £42.60 specifically: something in the shape of **74530045062001234567890**. That is the ledger line. It is the number the card networks use to talk about your lunch as *money that moved*, rather than as *a question that was asked*. If you dispute the charge, this is the number the dispute is filed under.

Now the useful part. Suppose you look at your banking app on Wednesday and see £42.60 twice.

If you ring the café and read out the counter number, they can do nothing with it, because every card machine they own has been through 004821 many times. If you read out the approval code, they can look, but approval codes are short and get reused, so they may find several. What actually settles the question is the twenty-three digit ledger reference, one per movement of money. Two of those means

two payments and one needs refunding. One of those, plus one leftover approval that never became a ledger line, means you were charged once and are looking at a hold that will fall off your statement in a few days.

Same event. Four numbers. Only one of them answers the question you actually asked.

Where the plain version stops being true

The wristband is not unique for ever, and it is not necessarily a number. The twelve-character reference is defined in the card message standard as twelve *alphanumeric* characters, not twelve digits. It can contain letters. It can begin with zeros. Any system that stores it as an integer will silently destroy it, and a great many systems do. It is also only unique within the organisation that issued it and within a time window that organisation chooses. Two different acquiring banks can and do produce the same twelve characters on the same day for two completely unrelated transactions. It is a good search key inside one estate. It is not a global identifier, and treating it as a primary key across parties is how reconciliation breaks.

The till slip code does not name the purchase. The six-character approval code says that an approval occurred. It does not encode the amount, the card, the merchant or the time. Issuers generate them cheaply and reuse them freely. Matching two records on approval code alone will produce false matches at any real volume, and it will produce them most often exactly when you are under pressure, because the customer with a problem is the one who has the receipt in front of them and nothing else. The approval code is a corroborating field. It is never a lookup key on its own.

The ledger reference is not the same journey. The twenty-three digit reference is created during clearing, which is a separate, later, batch-shaped process from the authorisation that happened at the till. An authorisation that is never cleared never gets one. A declined transaction never gets one. A hold that is left to expire never gets one. So there is a whole class of customer complaint, arguably the most common one, where the correct answer is "there is no ledger reference, because there is no charge", and the analogy of one wristband per child breaks down entirely: some children never went into the museum.

The numbers do not travel unchanged. The coach analogy implies one teacher counting. In reality the message crosses a terminal, a gateway, an acquirer's switch, the scheme's network, perhaps an issuer processor, and finally the issuer, and several of those nodes are entitled to replace the counter with one of their own before passing the message on, keeping a private map from the number they received to the number they sent. Your 004821 may be somebody else's 771903 two hops later. This is not a bug. It is how a switch keeps its own outstanding requests straight. It does mean there is no single number that is guaranteed to be present, unchanged, at every point in the chain, which is precisely the gap the card schemes later tried to close with identifiers of their own.

The technical version

The message these numbers live in

Card authorisation messages are ISO 8583 structures. The message type indicator is four digits: position one is the standard version (0 for the 1987 edition, 1 for 1993, 2 for 2003), position two the message class (1xx authorisation, 2xx financial, 3xx file actions, 4xx reversal and chargeback, 8xx network management), position three the function (0 request, 1 request response, 2 advice, 3 advice response) and position four the origin (0 acquirer, 2 issuer, 4 other). An authorisation request is therefore 0100, its

response 0110, a financial request 0200, a reversal request 0400 and a network management request 0800.

After the MTI comes the primary bitmap, sixty-four bits indicating whether data elements 1 to 64 are present. Bit 1, when set, signals a secondary bitmap covering elements 65 to 128. Everything discussed below is a data element addressed by that bitmap.

Four of them matter for tracing, and they are not interchangeable.

Element	Name	Format	Created by	Lives for
DE11	System trace audit number	n 6	The message originator	One request-response pair
DE37	Retrieval reference number	an 12	Usually the acquirer	Authorisation through to research
DE38	Authorisation identification response	an 6	The issuer	The life of the approval
DE31 (clearing)	Acquirer reference data / ARN	n 23	The acquirer, at clearing	The life of the money, and of any dispute

Two supporting elements do most of the disambiguating work and are frequently forgotten: DE41, card acceptor terminal identification, ans 8, and DE42, card acceptor identification code, ans 15. Alongside them sit DE7 transmission date and time, DE12 local transaction time (n 6, hhmmss), DE13 local transaction date (n 4, MMDD), DE4 amount, DE49 currency code as an ISO 4217 numeric code, DE39 response code (an 2) and DE55 for the EMV data.

Note DE13 carefully. In the 1987 catalogue the local transaction date is four digits, MMDD. There is no year in it. Every date-range query you write against a store of raw ISO 8583 must derive the year from somewhere else, normally DE7 or the acquirer's own receipt timestamp, and every year-end produces a fresh crop of reconciliation breaks in systems that did not.

DE11: the system trace audit number

DE11 is six numeric digits. It is assigned by whichever node originates the message, and it must be returned unchanged in the corresponding response so that the originator can pair the two. That is its whole function. It is a correlation key.

The consequences of that follow directly from the size of the field. Six digits gives you one million values before the counter wraps to zero and starts again. A busy terminal estate exhausts that in a day. A switch handling scheme traffic exhausts it in minutes. So a STAN is unique only within one originator, within one wrap cycle, and the sequence is very often per terminal rather than per estate, which means STAN 004821 exists simultaneously on every terminal you own.

This produces the single most common analytical error in card operations: searching for a STAN. A bare STAN search across an estate returns every terminal's 004821 for every day in the retention window. The correct key is composite. In practice you need the STAN together with the acquiring institution or terminal identity (DE41 or DE42), the local date (DE13, plus a year from DE7), and usually the amount (DE4) and currency (DE49) as a sanity check. Only then does the tuple identify one transaction, and even then only within one processing estate.

DE11 has one hard operational obligation attached to it. A reversal — a 0400 — must carry the STAN of the original message it is reversing, so that the issuer can find and release the hold it placed. Systems that generate a fresh STAN for the reversal and put the original nowhere in the message are the direct cause of authorisation holds that sit on customers' statements for a week. If you build a terminal application, store the original STAN in a durable local record before you send the request, not after you get the response, because the case in which you most need to reverse is the case in which no response arrived.

Mastercard's single-message system uses DE11 in exactly this way and names it accordingly. In Mastercard's fraud submission interface the identifier `TRC` is defined as "Trace Id (DE 11 in Single Message Authorization Message)", with a minimum and maximum length of 6 and a numeric data type. That is the STAN, promoted to an externally quotable identifier, and it is never quoted alone: Mastercard requires it in combination with `SER`, the Switch Serial Number, DE63 subfield 3, nine numeric digits. Two fields, because one is not enough. That is the standard telling you, in its own documentation, that a STAN is not a transaction identifier.

DE37: the retrieval reference number

DE37 is twelve alphanumeric characters, `an 12`. The name is the specification: it is the number you quote when you *retrieve* a transaction later, which historically meant asking the acquirer to pull the paper draft out of a filing cabinet.

Two things about it are routinely got wrong.

First, it is alphanumeric. Worldpay's published ISO 8583 reference guide records a change history entry to the effect that field 37 was corrected from numeric to alphanumeric, which tells you how easy the mistake is to make even inside a large processor. Store it as a fixed-width character field of exactly twelve. Do not trim it. Do not cast it. Leading zeros in a retrieval reference number are load-bearing.

Second, ISO 8583 does not specify any internal structure for DE37. It says twelve characters and stops. Everything you have read about what the characters mean is convention, and the convention belongs to whoever generated the value. Industry material describes one layout more often than any other, in which the leading characters carry a date and an hour, commonly a single digit of year, three digits of day-of-year and two digits of hour, with the remaining six characters a sequence that is often just the STAN. Acquirers do generate values of exactly that shape. But no published specification requires it, none of the public sources that set it out is a primary one, and so you must not write a parser that assumes it for traffic you did not generate. Some acquirers pack a terminal identifier in there instead. Some emit an opaque sequence. Some processors' own integration guides tell the merchant it may put whatever it likes in the field so long as the value comes back unchanged. A parser that assumes structure will one day interpret a letter as a month.

What DE37 buys you that DE11 does not is persistence. The retrieval reference number is generated once and carried forward, so the same value appears on the authorisation, on the advice, and typically on the acquirer's own clearing record and in the merchant's settlement report. That makes it the natural join key between the operational world of authorisations and the financial world of settlement, from the merchant's and acquirer's point of view. It is the best single number a merchant can hold, and it is the one your gateway should be surfacing to you in its API responses.

It is not, however, an identifier the issuer will necessarily accept as authoritative, and it is not unique across acquirers. Twelve characters and no registry of who may use which range means collisions between institutions are not merely possible but arithmetically inevitable at scheme volumes. Within your acquirer it identifies a transaction. Outside it, it is a strong hint.

India is the clearest illustration of DE37's second life. The retrieval reference number that appears in an IMPS or UPI transaction — the "UPI reference number" that Indian consumers are told to quote — is a twelve-digit RRN in the ISO 8583 lineage. Oracle's India IMPS payments documentation states plainly that the system "auto generates the RRN (Retrieval Reference Number) on authorization of the payment transaction" and that "this is a 12-digit number". The identifier a customer in Bengaluru reads off a UPI app screen and the identifier in DE37 of a card authorisation in Manchester are the same field, from the same standard, doing the same job.

DE38: the authorisation identification response

DE38 is six alphanumeric characters. It is generated by the issuer, or by the scheme acting on the issuer's behalf during stand-in processing, and it is present only when the transaction was approved. It travels back in the 0110 response alongside DE39, the response code, which in the 1987 catalogue is two alphanumeric characters, with `00` conventionally denoting approval. (The 2003 edition of the standard renames the field altogether: ISO's own published contents for ISO 8583-1:2003 list bit 39 as the action code, and dialects built on the later editions generally carry three-digit code values rather than two.)

DE38 is the number printed on the customer's receipt, labelled "auth code" or "approval code", which makes it the number you will most often be handed at the start of an investigation. It is also the weakest of the four.

Six alphanumeric characters is a small space. Issuers do not coordinate their generation, and there is no requirement that a given issuer's codes be unique across time, across cards or across merchants — only that the code be usable by the issuer to identify the approval it granted. The practical rule that follows is that an authorisation code is only meaningful in the presence of the card, the amount and the date. A support agent searching a transaction database on approval code alone, with no other predicate, is running a query that becomes less reliable as the business grows.

There are two further seams worth knowing. When an issuer is unavailable and the scheme approves on its behalf under stand-in processing, the approval code in DE38 was generated by the scheme, not the issuer, and the issuer may have no record of it until the advice reaches them. And in a partial approval, the amount approved is not the amount requested, so an approval code paired with the requested amount rather than the approved amount will not reconcile. DE38 tells you that a yes happened. DE4, DE49 and DE39 tell you what the yes was.

The acquirer reference number

The acquirer reference number is the one identifier in this chapter that is not created at the point of sale. It comes into existence during clearing, when the acquirer assembles the day's approved transactions into a settlement file and submits them to the scheme.

On Visa's side, the BASE II clearing system's own glossary defines it exactly: "Acquirer Reference Number: A 23-digit identification number associated with every draft and voucher. It consists of a Format Code, BASE Identification Number (BIN), Capture Date, Film Locator, and Check Digit." Five components, twenty-three digits, and the vocabulary tells you the history — a "film locator" is where the microfilm image of the paper sales draft was filed. The capture date is expressed as a Julian date, which Visa's glossary defines as "a date expressed as the day's position in a year rather than in a particular month. The format is YDDD or YYDDD."

On Mastercard's side the field is called Acquirer Reference Data and lives in DE31 of the clearing message. Mastercard's own developer documentation for fraud submission specifies the constraint

precisely: the identifier `ARN` is "Acquirer Reference Number (DE31 in Clearing Message)" and for it "Min Length is 23, Max Length is 23 and Data Type is N". Exactly twenty-three, numeric, no exceptions.

The check digit at the end is the practically useful part. It is computed from the preceding digits by a defined formula, which means a malformed ARN can be rejected before it costs anyone a research request. It also means that if you are validating ARNs you did not generate, you should validate the check digit rather than the semantics: do not try to parse the BIN and the date out of an ARN and reason about them, because the format code in the first position exists precisely to signal that the remaining layout may differ.

Two properties of the ARN follow from its birth at clearing rather than at authorisation, and both matter enormously in customer support.

The ARN exists if and only if the transaction cleared. No clearing, no ARN. Authorisations that expired, holds that were reversed, declines, and transactions still inside the acquirer's capture window all have no ARN and never will.

The ARN is the key the dispute process runs on. When an issuer initiates a chargeback, when an acquirer represents it, when a refund is traced across the network, the ARN is what identifies the item. It is also the number a merchant can give a customer that the customer's own bank can actually act on, which is why "please ask the merchant for the ARN" is the standard advice issuers give. It travels, in a way that a STAN never does.

The scheme's own key, and why it had to be invented

By this point the shape of the problem should be obvious. The acquirer owns a number. The issuer owns a number. The terminal owns a number. The clearing file owns a number. Nobody owns a number that is present, unchanged, at every hop, in both the authorisation and the clearing worlds. Card networks worked around this for decades with fuzzy matching on card number, amount, date and merchant.

Both major schemes eventually created an identifier of their own to close the gap.

Visa assigns a Transaction Identifier to each authorisation. As the Merchant Risk Council summarises it, it is "a numeric value up to 15 digits that uniquely identifies the original authorization", and the industry commonly calls it the Network Transaction Identifier or NTI. Its purpose is to let a later transaction point back at an earlier one — the classic case being a merchant-initiated recurring payment that must reference the original cardholder-authenticated authorisation.

Mastercard's equivalent is built from the Banknet reference number. In Mastercard's own field definitions, `BRN` is the "Banknet Reference Number (DE63, subfield 2 in Dual Message Authorization Message)", with a minimum length of 6 and maximum of 9, alphanumeric, and the documentation warns explicitly that "DE63, subfield 1 (Financial Network Code) is NOT to be appended". The composite identifier Mastercard asks merchants to store and echo — the Trace ID — is fifteen positions: positions 1 to 3 the Financial Network Code, positions 4 to 9 the Banknet Reference Number, positions 10 to 13 the settlement date as MMDD, and positions 14 to 15 blank.

Look at what that construction admits. To get an identifier that is stable enough to quote, Mastercard has to combine a network code, a reference number and a date. Three fields, because the reference number alone is six characters and six characters is not enough to be unique. This is the same lesson as the STAN, at a higher level of the stack: identity in payments is almost always composite, and any system that stores a single "transaction ID" column and believes it has captured identity is storing a correlation key and calling it a name.

The complete picture for one card payment

Putting it together for the £42.60 lunch in Leeds on Tuesday 3 March. These values are illustrative, not real.

What	Value	Who made it	Where it appears	Who can search on it
STAN (DE11)	`004821`	The terminal	0100 and 0110 only	Terminal estate operator, gateway, acquirer switch, each within its own hop
RRN (DE37)	`506201304821`	The acquirer	Authorisation, advice, clearing, merchant settlement report	Acquirer, gateway, merchant
Auth code (DE38)	`07B4C1`	The issuer	0110 response, customer receipt	Issuer, weakly
ARN (DE31 clearing)	`74530045062001234567890`	The acquirer, at clearing	Clearing file, dispute records, issuer research	Acquirer, scheme, issuer
Merchant order ID	`BC-2026-03-03-1187`	The merchant	The merchant's own database, sometimes the descriptor	The merchant only

That last row is the one merchants forget to design. Your own order identifier is the only number in the table you control, the only one guaranteed to be unique in your own system, and the only one you can attach to a customer record. Every investigation that goes well starts there and moves outward. Every investigation that goes badly starts with a number somebody read off a bank statement.

A customer says they were charged twice

Here is the order to work in, and why.

Step zero: establish whether there are two charges or one charge and one hold. This resolves the majority of cases and costs nothing. An authorisation that has not cleared appears in a banking app as a pending item and has no ARN. Ask the customer whether both lines are pending, or whether one is pending and one has posted, or whether both have posted. If both are pending and the amounts are identical, the likely cause is a double authorisation at the terminal or an unreversed retry, and the remedy is a reversal, not a refund. Refunding an uncleared authorisation creates a credit that settles while the hold quietly expires, and the customer ends up net up. This mistake is common and expensive.

Step one: collect what the customer actually has. Date, amount, currency, last four digits of the card, the descriptor as it appears on their statement, and the approval code if they have a receipt. Do not ask them for an ARN. Most issuers do not show it in the app, and asking is how a support conversation becomes a complaint.

Step two: find the order in your own system. Search on your merchant order identifier, or on amount plus date plus card token or last four. You are looking for how many order records exist and how many authorisation attempts each one made. This is where genuine double-submits, retried API calls without idempotency, and repeated tap events reveal themselves.

Step three: get the gateway or acquirer record for each attempt. For each attempt you now want DE37, the retrieval reference number, DE38, the approval code, DE39, the response code, and the local date and time. Two approved attempts with different retrieval reference numbers is two authorisations. Two approved attempts with different approval codes and different STANs one minute apart is almost always a terminal double-tap or a customer who tapped again after a slow response.

Step four: check clearing. For each authorisation, does an ARN exist? One ARN means one charge and the second authorisation is a hold. Two ARNs means two charges and one of them needs refunding. Zero ARNs on a same-day query means nothing has cleared yet and you should not conclude anything until the batch closes.

Step five: escalate with the right number to the right party. To your acquirer, quote the ARN if the item cleared and the retrieval reference number if it did not. To the scheme, via your acquirer, the ARN. To the issuer, if the customer is doing the asking, the ARN plus the date and amount. The approval code goes along as corroboration in all three cases and as the primary key in none of them.

What not to do at any step: do not search on STAN alone; do not match on approval code alone; do not assume the gateway's transaction identifier means anything to the acquirer, because it is the gateway's private key and typically does not appear in any ISO 8583 field at all; and do not tell the customer a number is "the transaction ID" when it is a correlation key that will be reused by teatime.

The underlying causes are a short list, and each has a signature. Two authorisations seconds apart with sequential STANs: terminal double-submit. Two authorisations minutes apart with different terminal identifiers: the customer tried a second till. An authorisation followed by a larger authorisation with the same retrieval reference lineage: an incremental authorisation, common in hotels and car hire, entirely legitimate. An original amount posting alongside a slightly larger cleared amount: a tip adjustment, where the cleared figure exceeds the authorised one within scheme tolerance. A charge and a matching credit landing days apart: a reversal that arrived after clearing rather than before, which looks like a double charge for as long as it takes the credit to post.

Bank transfers: the reference is not an identifier

Everything above concerns card rails. Account-to-account transfers have a parallel set of identifiers, and one important structural difference: the field the customer types is not the field the system traces on.

In the United Kingdom, a Bacs Direct Credit or Direct Debit is carried in a Standard 18 file. Open Banking Limited's mapping of its payment API onto the scheme formats, which cites the Bacs Electronic Funds Transfer file structures document PN5011 and its section on credit and debit payment instructions for bank grade users, lists the Standard 18 fields: field 1 destination sorting code, six characters; field 2 destination account number, eight; field 5 originating sorting code, six; field 6 originating account number, eight; field 8 amount in pence, eleven; field 10 service user's reference, eighteen; field 11 destination account name, eighteen.

That eighteen-character service user's reference is the entire space you have to say what a Bacs payment is for. It is what a building society roll number goes in. It is what an invoice number goes in. It is not unique, not validated, not guaranteed to reach any particular system at the far end, and if the sending customer types it wrong nothing anywhere will notice. Eighteen characters is also less than the thirty-five that ISO 20022 allows in a structured remittance reference, so a payment initiated through a modern API with a full-length reference must be either truncated or rejected at the boundary. Open Banking's specification says exactly that: it is for the account servicing institution "to decide whether to reject the payment-order consent or truncate the field".

Faster Payments is, perhaps surprisingly to anyone who has only worked on cards, also an ISO 8583 system — a different dialect, using the private-use data elements for its own purposes. Open Banking's mapping gives the field numbers: field 62 carries the end-to-end reference at 31 characters, field 120 the reference information at 18, field 121 the remittance information at 140, fields 42 and 43 the originating credit institution and customer account number, fields 95 and 35 the beneficiary equivalents, field 118 the beneficiary customer account name at 40, field 61.1 the payment sub-type and field 122 regulatory reporting. Pay.UK publishes an ISO 20022 standards library for FPS describing the recommended conversion of those ISO 8583 messages into ISO 20022.

Note what this does to your mental model of data element numbers. Data element 62 in a card authorisation is a private reserved field that Visa uses for its own purposes. Data element 62 in a Faster Payments message is the end-to-end reference. The number is not the meaning. The number plus the dialect is the meaning, and any library that decodes ISO 8583 without being told which specification it is decoding against is guessing.

CHAPS took the other road. It carried MT103 messages, in which the beneficiary reference sits in field 70 across up to four lines of thirty-five characters, with the code `/ROC/` introducing the ordering customer's reference and `/RFB/` introducing a reference for the beneficiary, the latter limited to sixteen characters in practice. In June 2023 the CHAPS high-value payment system migrated to ISO 20022, as the Bank of England records in its own policy statement on mandating enhanced data.

That migration matters here because ISO 20022 finally provides what card rails still lack. A pacs.008 customer credit transfer carries a payment identification block containing several distinct identifiers: an instruction identification, which is point-to-point between one pair of institutions; an end-to-end identification, which the debtor supplies and which must be passed unchanged all the way to the creditor; a transaction identification; and the UETR, the unique end-to-end transaction reference. The UETR is a thirty-six character UUID version 4 as defined by IETF RFC 4122, now superseded by RFC 9562. It is carried in field 121 of the user header block of an MT message and in the payment identification block of a pacs.008, and it has been mandatory on customer credit transfers over Swift's FIN network since 18 November 2018.

A UETR is what the card world does not have: one value, generated once, unchanged at every hop, unique by construction rather than by convention, and long enough that no two payments will ever collide. If you have wondered why a Swift payment can be tracked end to end and a card payment cannot, the answer is a hundred and twenty-eight bits of randomness against six digits of a wrapping counter.

India: UTR, RRN and what a customer can actually trace

India runs several rails and each stamps its own identifier, which is why the question "what is my UTR" has more than one correct answer.

For RTGS, the Reserve Bank of India's own FAQ is unambiguous: "Unique Transaction Reference (UTR) number is a 22-character code used to uniquely identify a transaction in RTGS system." Twenty-two characters, alphanumeric, generated by the sending bank. The RBI is equally clear about who may use it, noting that customers do not have the facility to track the transaction themselves — the UTR is what the bank quotes to the central bank when it asks about status on the customer's behalf.

Indian payments material consistently describes a decomposition in which the first four characters are the sending bank's IFSC prefix, followed by a character denoting the RTGS system, a channel or message-type character, the date, and a sequence. That description is convention rather than published rule: the RBI's FAQ gives the length and says nothing about the layout, no primary specification of

the internal structure is public, and you should not parse it. NEFT references are commonly quoted as sixteen characters with a similar bank-code-then-date-then-sequence shape, again by convention rather than by published rule.

For IMPS and UPI the identifier consumers are told to quote is the twelve-digit retrieval reference number. It is the same DE37 discussed above, because the National Payments Corporation of India's switch is ISO 8583 derived. A UPI payment additionally carries a transaction identifier assigned by the payment service provider, which is the identifier your app uses internally and which the receiving bank generally cannot resolve.

The practical guidance for anyone building support processes for Indian payments is therefore the mirror image of the card guidance. The twelve-digit RRN is the number both banks can see. The PSP's transaction identifier is the number only your own systems can see. The UTR is the number that exists for RTGS and NEFT and not for UPI. And in every case the RBI's structural point holds: the identifier is an input to a request made by a bank, not a self-service tracking code. When a customer says "I have the UTR, why can't you find it", the honest answer is that the number identifies the payment inside a system they do not have access to, and so do you unless you are a bank.

Design rules that follow

Store every identifier you are given, and never overwrite one with another. The STAN, the retrieval reference number, the approval code, the ARN when it arrives, the scheme transaction identifier, the gateway's own key and your own order identifier are seven different columns, not one. The cost of seven columns is nothing. The cost of having only the one you thought you needed is a research request per customer.

Store them as text, fixed width, with the specification's width. `an 12` means twelve characters, letters permitted, leading zeros significant. `n 23` means twenty-three digits, and twenty-three digits will not fit in a 64-bit integer.

Make every lookup key composite by default. STAN plus terminal plus date plus amount. Retrieval reference number plus acquirer. Approval code plus card plus amount plus date. If a query in your support tooling takes a single identifier and returns a single row, ask which identifier it is, because for three of the four in this chapter that query is lying to you.

Generate your own identifier at the earliest possible moment, before the first network call, and put it everywhere you are allowed to. Where the rail gives you a durable customer-visible reference field — the eighteen characters of a Bacs service user's reference, the thirty-five of an ISO 20022 end-to-end identification — spend it on something you can look up, not on a description of the goods.

And be careful what you tell customers a number means. The one on their receipt is an approval code and it proves that a bank said yes. The one that proves money moved is created hours later by a different institution, and until it exists there is nothing to trace. Both of those statements are true simultaneously, and explaining the difference clearly is most of the job.

28.98 Common wrong ideas

Wrong: the retrieval reference number is twelve digits. **Right:** it is `an 12`, twelve alphanumeric characters whose leading zeros are load-bearing, and any system that stores it as an integer silently destroys it.

Wrong: the leading characters of a retrieval reference number encode a year, a day and an hour, so you can parse them. **Right:** ISO 8583 specifies no internal structure at all; the common layout is a

convention belonging to whoever generated the value, and some acquirers pack a terminal identifier or an opaque sequence in there instead.

Wrong: the approval code identifies the purchase. **Right:** six alphanumeric characters generated without coordination between issuers say only that a yes happened, and matching on them alone produces false matches at any real volume.

Wrong: the system trace audit number is the transaction identifier. **Right:** six digits wrap at a million, the sequence is very often per terminal, and Mastercard's own documentation requires the trace identifier to be quoted alongside the switch serial number because one is not enough.

Wrong: a reversal can carry a fresh trace number. **Right:** a `0400` must quote the original message's trace number so the issuer can find and release the hold, and failing to do so is the direct cause of authorisation holds sitting on customers' statements for a week.

Wrong: every transaction has an acquirer reference number. **Right:** the ARN is created during clearing, so declines, expired authorisations, reversed holds and anything still inside the capture window have none and never will, and "there is no reference because there is no charge" is often the correct answer.

Wrong: you can parse the bank identification number and the capture date out of an ARN and reason about them. **Right:** the format code in the first position exists precisely to signal that the remaining layout may differ, so validate the check digit rather than the semantics.

Wrong: data element 62 means the same thing in every ISO 8583 message. **Right:** it is a private reserved field on card rails and the end-to-end reference in Faster Payments; the number plus the dialect is the meaning, and a decoder not told which specification it is decoding against is guessing.

Wrong: the reference a customer types into a bank transfer is an identifier the system traces on. **Right:** the eighteen characters of a Bacs service user's reference are unvalidated free text, shorter than the thirty-five ISO 20022 allows, and the receiving institution may truncate or reject them.

Wrong: a UTR is a self-service tracking code and every Indian payment has one. **Right:** RTGS has a twenty-two character UTR that a bank quotes on the customer's behalf, IMPS and UPI use the twelve-digit retrieval reference number, and a UPI payment's own service-provider identifier is visible only inside that provider.

28.99 Chapter summary in 20 lines

1. A card payment leaves behind four numbers with four owners, four lifetimes and four jobs, and asking the wrong party for the wrong one gets you a shrug.
2. The system trace audit number in DE11 is six digits assigned by whichever node originates a message and returned unchanged so that the originator can pair the answer with the question.
3. Six digits wrap at a million and the counter is often per terminal, so the same trace number exists simultaneously on every terminal in an estate and was somebody else's transaction yesterday.
4. Searching on a trace number alone is the commonest analytical error in card operations; the key must be composite, adding terminal or acquirer identity, the local date, and usually the amount and currency.
5. A reversal must quote the original trace number, and because the case in which you most need to reverse is the case in which no response arrived, the original must be stored durably before the request is sent.

6. The retrieval reference number in DE37 is twelve alphanumeric characters created by the acquirer and carried forward into the advice, the clearing record and the merchant's settlement report.
7. Its leading characters often look like a year, a day of the year and an hour, but no published specification requires that layout, and a parser assuming it will one day interpret a letter as a month.
8. DE37 is the best single number a merchant can hold, but it is unique only within one acquirer, and collisions between institutions are arithmetically inevitable at scheme volumes.
9. The same field has a second life in India, where the twelve-digit retrieval reference number is what an IMPS or UPI customer is told to quote.
10. The authorisation identification response in DE38 is six alphanumeric characters generated by the issuer, or by the scheme standing in for it, and only when the answer was yes.
11. It is the number printed on the receipt and therefore the one a customer hands you first, and it is also the weakest of the four, encoding nothing about the amount, the card, the merchant or the time.
12. The acquirer reference number is not created at the point of sale at all: it is born during clearing, twenty-three digits made of a format code, a bank identification number, a capture date, a film locator and a check digit.
13. Validate that check digit rather than the semantics, because the format code exists to warn that the rest of the layout may differ.
14. The ARN exists if and only if the transaction cleared, and it is the key on which the entire dispute process runs, which is why issuers tell customers to ask the merchant for it.
15. Nobody owns a number that is present, unchanged, at every hop, which is why Visa created a Transaction Identifier and Mastercard assembled a fifteen-position Trace ID from a network code, a Banknet reference and a settlement date.
16. That construction admits the general rule: identity in payments is almost always composite, and a system with one "transaction ID" column is storing a correlation key and calling it a name.
17. When a customer reports being charged twice, establish first whether there are two charges or one charge and one hold, because refunding an uncleared authorisation creates a credit that settles while the hold quietly expires.
18. Then work outward from your own order identifier, the one number you control, through the gateway or acquirer record for each attempt, through clearing, and only then escalate with the right number to the right party.
19. Account-to-account rails have the same problem in different clothes, from an eighteen-character Bacs reference that nothing validates to a Faster Payments dialect in which data element 62 means something entirely different from what it means on card rails.
20. ISO 20022 and Swift's UETR are what the card world lacks — one value, generated once, unchanged at every hop, unique by construction — which is the whole reason a Swift payment can be tracked end to end and a card payment cannot.

Sources: ISO 8583 data element definitions; Visa BASE II Clearing System Overview glossary; Mastercard Developers field definitions for ARN (DE31), Banknet Reference Number (DE63.2), Trace ID (DE11) and Switch Serial Number (DE63.3); Merchant Risk Council guidance on Visa Transaction Identifier and Mastercard Trace ID; Worldpay ISO 8583 Reference Guide; Open Banking Limited Domestic Payment Message Formats (FPS ISO 8583, Bacs Standard 18 per Bacs PN5011, CHAPS MT103); Pay.UK Faster Payment System ISO 20022 standards library; Bank of England policy statements on ISO 20022 in CHAPS; Reserve Bank of India RTGS FAQ; Oracle India IMPS Payments User Guide; IETF RFC 9562.

Response Codes

29.0 What this chapter gives you

1. You will be able to point to the two-character field that carries the entire verdict of a card authorisation, name it as DE39, and say why a database column typed as an integer is a defect waiting for a Tuesday.
2. You will be able to explain why an issuer refuses to tell a merchant the reason for a decline, and show with arithmetic that a fully specific vocabulary would let an attacker take a bare card number to a working expiry, security code and liveness check in about a thousand probes.
3. You will be able to take a batch of failed subscription charges and split it into the group that deserves a retry timed around payday, the group that needs a new card number, and the group that must never be attempted again.
4. You will be able to name the four separate mechanisms by which blind retrying damages a business, only one of which shows up in the conversion number.
5. You will be able to identify the values that are approvals despite not being `00` — partial approval, VIP approval, honour with identification, no reason to decline — and say what a merchant throws away by treating them as failures.
6. You will be able to read a Mastercard decline as a pair, the response code plus the merchant advice code, and follow it to the correct next action rather than to a queue.
7. You will be able to place a Visa decline in one of the four decline categories and explain why the most common code of all sits in the generic bucket.
8. You will be able to explain why the same two characters mean a PIN verification problem on one network, a fraud decline on another and a token restriction on a third, and why storing the code without the scheme destroys the information.
9. You will be able to distinguish a timeout from a decline and send a reversal carrying the original trace number instead of a second identical request.
10. You will be able to recognise a strong customer authentication soft decline and route it through EMV 3DS rather than retrying it or losing the sale.

The plain version

Think about a door with a bouncer on it.

You want to get into a club. The bouncer looks at you, looks at your ID, thinks for a second or two, and gives you one of two kinds of answer. The first kind is “in you go”. There is only one way to say that. The second kind is “not tonight” — and behind those two words sit fifty different reasons, and the bouncer will not tell you which one applies to you.

He knows perfectly well. Maybe you are too young. Maybe your name is on a list behind the counter. Maybe the fire officer capped the building at three hundred people and you are number three hundred and one. Maybe he does not like the look of the four people standing behind you and has decided to break up the group by refusing the one at the front. But what he says is "not tonight", in the same flat voice, every time.

This is not laziness and it is not bad manners. It is the job. If the bouncer said "your ID is fake, the hologram is wrong", the next person along would bring a better hologram. If he said "you are on the barred list", the barred person would learn to send a friend in first to test the water. Every extra word he gives away is a free lesson for the next person who wants in and should not get in. So the club has a rule: one way to say yes, one flat phrase for no, and the real reason stays inside the building.

Now notice something else about "not tonight". It comes in two flavours that sound identical.

One flavour is temporary. We are full. Come back at midnight and there will be room. The door has not shut on you personally; it has shut on this moment.

The other flavour is permanent. You are barred. You were barred in January and you will be barred in December. Coming back at midnight achieves nothing, and the only thing that will change your position is a conversation with the manager during the day.

And here is the thing that catches people out. If you cannot tell the two apart, the sensible-looking strategy is to keep trying. So you come back at ten past, and twenty past, and half past, and by eleven o'clock the bouncer has stopped seeing a customer and started seeing a problem. You are no longer being refused because the club is full. You are being refused because you are the man who keeps coming back. Your own persistence has converted a temporary no into a permanent one.

A card payment works exactly like this, and the whole thing turns on a box in the reply message that holds two characters. When a card is tapped, a message goes out asking whether this card may be charged nine pounds ninety-nine. A message comes back, and near the front of that reply there is a small fixed slot, two characters wide, carrying the entire answer. If it contains `00`, the answer is yes. If it contains anything else at all, the answer is no. There are about sixty values that mean no, and the difference between a payments operation that works and one that quietly bleeds money is almost entirely in what you do with those two characters.

Take a real example. Ravinder runs a gym in Coventry. He has 1,000 members paying £24.99 a month, which is £24,990 a month if everybody pays, and he collects it all on the first of the month by charging the card each member has left on file. On 1 March he sends 1,000 requests. Nine hundred and forty-seven come back `00`. Fifty-three do not, and this is roughly how they break down:

Code	What it means, roughly	How many	Money
51	Not enough money in the account	31	£774.69
05	Do not honour — no reason given	11	£274.89
54	The card has expired	6	£149.94
41 or 43	Card reported lost or stolen	3	£74.97
91	The member's bank did not answer	2	£49.98
		53	£1,324.47

Now, the wrong thing to do, and the thing an awful lot of software does, is to treat all fifty-three the same. Put them in a queue. Try each of them again four times a day for a week. That is 53 x 4 x 7 = 1,484 additional attempts. It will recover something — a handful of the empty accounts will fill

up when wages land — but it will recover perhaps twenty of them. The other 1,464 attempts achieve nothing at all except to make Ravinder's gym look, to a computer at each member's bank, exactly like somebody testing stolen card numbers. Some of those attempts carry a fee. Some of them make the *next* attempt more likely to be refused.

The right thing to do is to read the two characters and treat the five groups as five different problems.

The thirty-one members with code 51 have a working card and an empty account. Nothing is wrong with the card, the merchant or the amount. What is wrong is the date. Retrying on the second of the month is close to pointless; retrying on the fourth, after most people have been paid, is not. Two attempts, spaced out, will typically bring back most of them. Call it twenty-four of thirty-one, £599.76 recovered, at a cost of sixty-two attempts rather than 868.

The six members with code 54 have a card that expired. There is no number of retries that unexpires a card. The only two things that work are asking the member for a new one, or using the card networks' own updating service, which quietly hands merchants the replacement card number when a bank reissues. Retrying is not merely useless here, it delays the one action that would have worked.

The three members with codes 41 and 43 have had their cards reported lost or stolen. That card is dead. It will be dead tomorrow and dead in a year, and both card networks will charge Ravinder for continuing to ask.

The two members with code 91 did not get an answer because their bank's computer was not responding. This one genuinely does deserve an immediate retry. It is the only code in the list where "just try again" is the whole correct answer.

And then there are the eleven with code 05, do not honour, which is the bouncer's flat voice. Do not honour means the member's bank has decided not to pay and is not going to say why. It might be a closed account, a fraud score, or the member having told their bank to stop this subscription and the bank doing as it was asked. Ravinder cannot tell, and — this is the important part — he is not supposed to be able to tell. The correct handling is one further attempt after several days, and if it comes back 05 again, an email to the member. Not a queue.

Total: about ninety extra attempts instead of 1,484, and about £800 of the £1,324 recovered instead of about £500. Same fifty-three customers. The only difference is that somebody read the two characters.

That is the plain version. One value means yes, everything else means no, and the no is deliberately vague because a specific no is a free lesson for a criminal. There are two kinds of no, temporary and permanent, and they look identical from the outside, which is precisely why blind retrying is so tempting and so expensive.

Where the plain version stops being true

The bouncer is not always the bank, and often nobody's bank was asked. The analogy has one door and one man on it. A real card transaction can be refused at five different points, and the two-character code you eventually see does not say which. The chip in the card can decline the transaction by itself, offline, before any message goes anywhere. The terminal or the point-of-sale application can refuse it on a floor limit or a configuration rule. The merchant's own fraud screening can block it before it ever reaches the acquirer. The acquirer or its gateway can reject it for a malformed field, an unsupported currency, or a merchant-level velocity rule. The card scheme itself can answer on the issuer's behalf when the issuer's systems are unreachable, using parameters the issuer lodged in advance — Visa calls this stand-in processing. Only after all of those does the actual issuer get a vote. So "your bank

declined this" is a sentence customer service staff say many times a day and which is frequently untrue. Worse, several of those layers write into the same field using the same vocabulary, so the shape of the answer tells you nothing about its author.

"00 is yes and everything else is no" is a good rule with real exceptions, and you must know them. Code 10 is a partial approval: the issuer has approved a smaller amount than you asked for, and if your software treats that as a decline you have just thrown away a sale you had already won, and possibly left a hold on the customer's account for money you will never take. Code 11 is an approval. Code 08, honour with identification, is an approval conditional on the merchant checking something. Code 85 is not a decline at all — it is the answer to a question you asked without wanting money, such as "is this card real and does this postcode match", and it means "no reason to decline". And `00` in the response to a *reversal* means the reversal was accepted, which tells you nothing whatever about whether the original transaction was approved. Meanwhile `00` on an authorisation does not mean money has moved. It means a bank has set some of its customer's money aside and promised to pay if you ask properly. Those are different facts and they live in different chapters.

The vagueness is being unwound on purpose, but only in one direction. The plain version says the reason stays inside the building. That was true for thirty years and it is becoming less true, because the card schemes worked out that total opacity was costing everybody money in failed recurring payments. Mastercard now requires issuers to split their generic declines into three buckets and to attach a separate advice field telling the merchant what to do next. Visa now classifies every decline code into one of four categories and penalises merchants who ignore the classification. But look carefully at what the schemes added. They added *what to do*, not *why*. "Do not try again" is a much more useful message than "do not honour", and it still does not tell the merchant that the account was closed on 3 March because the customer died. The direction of the disclosure is deliberate: the merchant learns enough to stop wasting attempts and not one thing more about the cardholder. The reason itself is still available, but only to the cardholder, from their own bank, over a channel where the bank has already checked who it is talking to.

Soft and hard are not properties of the code. Nowhere in the international standard that defines this field is there any concept of a soft decline or a hard decline. Those are operational categories invented afterwards by the card schemes, published in rulebooks that are revised on an annual cycle, and they differ between schemes, differ between regions, and move. A code that sits in the never-retry bucket this year may be moved to the retry-later bucket next year by a bulletin. More disconcertingly, the same two characters mean genuinely different things on different networks: code 79 means "already reversed by the switch" on Visa and "life cycle" on Mastercard, which are not remotely the same event. Any system that stores a decline as a normalised boolean, or that stores the code without also storing which scheme produced it, has destroyed the information it will need later.

The technical version

Where the field sits

Card authorisation messages are ISO 8583 structures. The message type indicator is four digits: position one gives the version of the standard (0 for the 1987 edition, 1 for 1993, 2 for 2003), position two the message class (1xx authorisation, 2xx financial, 3xx file actions, 4xx reversal and chargeback, 8xx network management), position three the function (0 request, 1 request response, 2 advice, 3 advice response), and position four the origin (0 acquirer, 2 issuer, 4 other). An authorisation request is 0100 and its response 0110; a financial request is 0200 and its response 0210; a reversal request is 0400 and its response 0410; a network management request is 0800 and its response 0810.

After the MTI comes the primary bitmap, sixty-four bits marking the presence of data elements 1 to 64, with bit 1 signalling a secondary bitmap covering 65 to 128. Data element 39, the response code, is an 2 in the 1987 catalogue: exactly two alphanumeric characters, fixed length, no more and no less. It is present in response messages — 0110, 0210, 0410, 0430, 0810 — and absent from requests.

The elements it must be read alongside are these.

Element	Format	Name	Why it matters here
DE4	n 12	Amount, transaction	On a partial approval the response amount is not the request amount
DE11	n 6	System trace audit number	Pairs the response to the request
DE38	an 6	Authorisation identification response	Present on approval only
DE39	an 2	Response code	The answer
DE44	an ..25	Additional response data	Where address and card-verification results are commonly carried
DE48	an ...999	Additional data, private	Where Mastercard carries the merchant advice code
DE49	n 3	Currency code, transaction, per ISO 4217	An approval is an approval of an amount in a currency
DE54	an ...120	Additional amounts	Balances and remaining amounts on some approvals
DE55	ans ...999	Integrated circuit card data	The chip's own verdict travels here

Two implementation notes follow immediately from that table and are worth stating flatly. DE38 is populated only when the transaction is approved, so an implementation that decides "approved" by checking whether an authorisation code is present is doing the right thing for the wrong reason and will eventually meet an issuer that populates DE38 with `000000` on a decline. DE39 is the authoritative field; everything else is corroboration. And DE39 is two characters, not two digits: values such as `1A`, `N7`, `R0`, `Q1`, `Z3`, `5C` and `9G` are all in live use, so a column typed as an integer is a defect waiting for a Tuesday.

The 1987 catalogue

Almost all card scheme traffic still uses the two-character response codes of the 1987 edition, and the base catalogue is worth having in front of you. It is grouped by intent rather than numerically, which is why the numbering looks arbitrary.

Code	ISO 8583:1987 meaning
00	Approved or completed successfully
01	Refer to card issuer
03	Invalid merchant
04	Pick-up

Code	ISO 8583:1987 meaning
05	Do not honor
06	Error
07	Pick-up card, special condition
08	Honour with identification
10	Approved for partial amount
11	Approved (VIP)
12	Invalid transaction
13	Invalid amount
14	Invalid card number (no such number)
15	No such issuer
19	Re-enter transaction
30	Format error
33	Expired card
34	Suspected fraud
38	Allowable PIN tries exceeded
41	Lost card
43	Stolen card, pick-up
51	Not sufficient funds
54	Expired card
55	Incorrect personal identification number
57	Transaction not permitted to cardholder
59	Suspected fraud
61	Exceeds withdrawal amount limit
62	Restricted card
65	Exceeds withdrawal frequency limit
75	Allowable number of PIN tries exceeded
91	Issuer or switch is inoperative
93	Transaction cannot be completed. Violation of law
94	Duplicate transmission
96	System malfunction

The catalogue reserves 45 to 50 and 69 to 74 for ISO use, 97 to 99 for national use, and the ranges beginning with a letter for national and private use, which is where every scheme's proprietary values live. Note the duplication the standard itself contains: 33 and 54 both say expired card, 34 and 59 both say suspected fraud, 38 and 75 both say PIN tries exceeded. The difference historically was whether the card was to be captured. That distinction is dead and the duplication survives, which is one small demonstration that this vocabulary is an archaeological site rather than a design.

The 1993 renumbering, and why you have probably never seen it

The 1993 revision of ISO 8583 replaced the two-character response code with a three-digit action code, still in DE39, and gave it a genuinely rational structure: the range says what happened, the value says why.

Range	Meaning
000-099	Approved
100-199	Denied, card not to be captured
200-299	Denied, card to be captured

Range	Meaning
300-399	Result of a file action
400-499	Result of a reversal or chargeback
500-599	Result of a reconciliation
600-699	Result of an administrative or retrieval request
700-799	Fee collection results
800-899	Network management results
900-999	Could not be processed, or liability transfer rejected

Inside those ranges the values are systematic. 000 is approved, 002 approved for partial amount, 007 approved and update the chip. 100 is do not honour, 101 expired card, 116 not sufficient funds, 117 incorrect PIN. The 200 range is the same list again but with the card to be captured: 200 do not honour, 208 lost card, 209 stolen card. And 907 is card issuer or switch inoperative, 911 card issuer timed out, 912 card issuer unavailable — three distinct failures that the 1987 catalogue collapses into 91.

This is a better design in every respect and it is not what the card networks run. Visa, Mastercard, American Express and Discover all froze their dialects on the 1987 shape and then extended it privately, so the 1993 action codes appear mainly in domestic switches, ATM networks and newer national schemes. A practitioner must know which one is on the wire before writing any comparison, because `10` and `100` are an approval and a decline respectively and are distinguishable only by knowing the version.

The published Visa set

Visa publishes its action code list openly on Visa Developer, and it is the closest thing to an authoritative reference for what a merchant will actually receive on Visa rails. The list includes the ISO base plus a substantial private extension. Some entries that matter:

Code	Visa's published description
00	Approved and completed successfully
04	Pick up card (no fraud)
05	Do not honor
07	Pick up card, special condition (fraud account)
10	Partial approval
14	Invalid account number (no such number)
41	Lost card, pick up (fraud account)
46	Closed account
54	Expired card or expiration date is missing
57	Transaction not permitted to cardholder
61	Exceeds approval amount limit
62	Restricted card (card invalid in this region or country)
78	"Blocked, first used" — Transaction from new cardholder, and card not properly unblocked
79	Already reversed (by Switch)
80	No financial impact
82	Negative CAM, dCVV, iCVV, or CVV results
85	No reason to decline a request for address verification, CVV2 verification, or a credit voucher or merchandise return

Code	Visa's published description
91	Issuer or switch inoperative and STIP not applicable or not available for this transaction; Time-out when no stand-in
94	Request is identified as a duplicate
1A	Additional customer authentication required
N0	Force STIP
N7	Decline for CVV2 failure
R0	Stop this Payment
R1	Stop all future Payments
R3	Stop all Merchants
Z3	Unable to go online; offline-declined
5C	Transaction not supported/blocked by issuer
9G	Blocked by cardholder/contact cardholder

Several of these repay attention. `R0`, `R1` and `R3` are the cardholder speaking through their bank: the customer has instructed the issuer to stop this payment, all future payments to this merchant, or all merchant-initiated payments respectively. A merchant that retries an `R1` is not fighting a machine, it is overriding an instruction its customer gave to their own bank, and both the scheme rules and consumer protection law take a dim view of it. `85` is the affirmative answer to a zero-amount verification, which is how a merchant checks that a card is real and that a billing address matches without taking any money. `80`, "no financial impact", is what a reversal response carries when the original was declined anyway.

Why the issuer is deliberately vague

The vagueness of `05` is not a documentation gap that somebody will fix. It is a control, and the argument for it is arithmetic.

Consider an attacker holding a list of 10,000 candidate card numbers of unknown quality, obtained from a breach, generated against a known issuer range, or scraped. They want to convert that list into a smaller list of numbers that can actually be charged. Now suppose issuers returned a fully specific decline for every case. Watch what the attacker learns from a single one-pound probe against a single card:

A response of 14, invalid account number, tells them the number does not exist. Cross it off.

A response of 54, expired card, tells them something far more valuable: the number *does* exist and their expiry date is wrong. An expiry is a month and a year, and cards are typically issued for three to five years, so a complete search of the plausible space is at most sixty combinations.

A response of N7 or 82, a card verification failure, tells them the number and expiry are both correct and only the three-digit security code is wrong. That is a thousand probes, with the response code confirming the moment they hit it.

A response of 51, not sufficient funds, tells them everything is correct: number, expiry, security code, and a live account that is merely empty at this instant.

So a fully specific decline vocabulary converts an unstructured guess into a staged search with a feedback signal at every stage: roughly 1,061 probes to take a bare card number and establish its expiry, its security code and its liveness. That is the definition of an oracle, and building one into the response field of the world's payment network would be an extraordinary thing to do.

Collapse all of it to `05` and the attacker learns exactly one bit per probe: not this one, for some reason, at this moment. The staged search collapses. This is why issuers' risk engines map dozens of distinct internal outcomes — negative file hit, velocity breach, device reputation, machine-learning score above threshold, account under investigation, merchant category blocked by the cardholder, sanctions screening alert, a mismatch the issuer would rather not name — onto a single unhelpful value.

There is a second reason, and it is legal rather than technical. Some declines cannot be explained to a merchant without breaking the law. If a payment is stopped because a sanctions or anti-money-laundering screen fired, telling the merchant why can constitute tipping off. Code 93, "transaction cannot be completed, violation of law", is about as much as anyone will say, and many issuers will not even say that much and will send `05` instead. Similarly, where the decline is about the *merchant* rather than the cardholder — an issuer that has decided to block a whole category, or a specific acceptor, because of chargeback history — an explicit code would amount to publishing a commercial judgement about a business to that business.

The disclosure is therefore directional rather than absent. The cardholder can find out. They ring the number on the back of the card, the bank authenticates them, and the bank tells them exactly what happened, because it is their money and their account. The merchant cannot, because the merchant is not the customer and is, statistically, sometimes the attacker. The right thing for a merchant to say to a customer holding a declined card is not "your bank rejected it" but "your bank has not told us why; if you ring them they will tell you".

Soft and hard, in the schemes' own words

Because "do not honour" wrecked recurring billing, both major schemes built classification systems on top of the code set. They are not the same system.

Visa assigns every decline code to one of four decline categories:

Category	Meaning	Merchant action
1	The issuer will never approve	Never reattempt with the same account details
2	The issuer cannot approve at this time	Wait, then reattempt within limits
3	The issuer cannot approve based on the details provided	Correct the data, then reattempt within limits
4	Generic response	Treated like category 2 for retry purposes

Category 1 is populated with the codes where the account or the card is simply gone: 04, 07, 12, 14, 15, 41, 43, 46, 57, and the stop-payment family R0, R1 and R3. Category 2 holds the temporary conditions: 03, 19, 51, 59, 61, 62, 65, 75, 78, 86, 91, 93, 96, N3, N4, 5C, 9G. Category 3 holds the data problems, which is where the retry actually has to change something before it is sent: 54, 55, 70, 82, 1A, 6P, N7. Category 4 is everything else, and — this is the sharp part — `05`, do not honour, sits in category 4. The generic code is in the generic category, and Visa has been pressing issuers to reduce their use of it in favour of a code from categories 1 to 3, because the whole scheme is worth less when the biggest single bucket is unclassifiable.

Mastercard went the other way. Rather than classifying the existing codes, it introduced three replacement values in DE39 and a separate advice field. Mastercard's own developer documentation defines

them plainly, noting that the response code "is passed to Mastercard in Data Element (DE) 39 of the issuer's network response message":

Code	Mastercard meaning
05	Do not honor
79	Life cycle (Mastercard use only)
82	Policy (Mastercard use only)
83	Fraud/Security (Mastercard use only)

So a Mastercard decline that would once have arrived as 14 or 54 now arrives as 79, life cycle, meaning something about the credential itself has changed. A decline that reflects an issuer rule arrives as 82, policy. A decline that reflects a fraud judgement arrives as 83. And alongside it, in DE 48 subelement 84, comes the merchant advice code, which is where the actual instruction lives.

MAC	Meaning
01	New account information available
02	Cannot approve at this time, try again later
03	Do not try again
04	Token requirements not fulfilled for this token type
05	Negotiated value not approved
21	Payment cancellation
24	Retry after 1 hour
25	Retry after 24 hours
26	Retry after 2 days
27	Retry after 4 days
28	Retry after 6 days
29	Retry after 8 days
30	Retry after 10 days
40	Consumer non-reloadable prepaid card
41	Consumer single-use virtual card number

Codes 24 through 30 are the most quietly remarkable thing here: the issuer telling the merchant, in the decline message itself, when the account is likely to have money in it. That is the answer to code 51 delivered as data instead of guesswork.

The Mastercard Switch Rules set out the intended pairings directly:

DE39	DE 48 subelement 84	Merchant advice	Merchant action
79 or 82	01	Updated information needed	Updated information found to be available in Mastercard ABU database; secure new information before reattempting
79 or 82	03	Do not try again	Updated credentials were not found to be available in Mastercard ABU database; do not retry

DE39	DE 48 subelement 84	Merchant advice	Merchant action
83	01	Additional information needed	Ensure card information is correct; authentication may improve likelihood of an approval — try using authentication such as EMV 3DS
83	03	Do not try again	Suspected fraud — do not retry
79, 82 or 83	02	Try again later	Retry transaction later

Read that table as a decision tree and the design becomes obvious. `79` or `82` with MAC `01` means go to the Account Billing Updater and get the new card number. `83` with MAC `01` means the credential is probably fine but the transaction was not authenticated, so run it through EMV 3DS and come back. Anything with MAC `03` means stop, permanently. MAC `21`, payment cancellation, means the cardholder has cancelled the recurring arrangement and the merchant's next move is a conversation, not a request.

Retry logic, and the four ways blind retries make it worse

Practitioners talk about retries as though the only question were how many attempts convert. There are four separate mechanisms by which unstructured retrying damages a business, and only one of them is visible in the conversion number.

The schemes charge for it. Visa operates an Excessive Reattempts rule under which a reattempt of a transaction that returned a category 1 decline is chargeable from the very first reattempt, and reattempts beyond a threshold within a rolling thirty-day window are chargeable for categories 2 and 3. The commonly published threshold is fifteen reattempts in thirty days, on the same account number at the same merchant, with the fee applying from the sixteenth; some acquirers publish the limit as twenty, and both figures appear in current merchant-facing guidance, which tells you that the rule has been revised and that regional variants exist. Mastercard runs an equivalent programme within Transaction Processing Excellence, assessing a fee for authorisation attempts beyond a threshold of previously declined attempts on the same account number, at the same card acceptor, and for the same amount, with published thresholds around ten declined attempts in twenty-four hours or thirty-five in a rolling thirty days, again with explicit regional variation. Mastercard additionally charges where a merchant retries a transaction that was declined with MAC `03` or MAC `21` and receives another `03` or `21`. The per-attempt amounts sit in the schemes' own fee schedules, which are not published, so no primary document will confirm a figure; what can be checked is what the acquirers and payment service providers who pass the charge on tell their merchants, and they quote it consistently. Adyen's published scheme fee documentation, to take one, puts the Mastercard excessive authorisations fee at USD 0.30 in the United States rising to USD 0.50 from the first of January 2025, EUR 0.55 in Europe, and USD 0.50 across Asia-Pacific, Canada and Latin America, and notes that the permitted retry counts themselves vary by region. The engineering conclusion does not depend on the figure: cap attempts below the lowest published threshold in any region you operate in, count them per account number per merchant per thirty days, and treat category 1 and MAC 03 as absolute.

The issuer scores you. Every authorisation request a merchant sends is a data point in the issuer's model of that merchant. A retry pattern of two attempts spaced four days apart looks like a subscription business. Four attempts a day for a week against cards that keep saying no looks like card

testing, because that is precisely what card testing looks like. Issuers respond by declining more of that merchant's traffic, including the good traffic, and the merchant discovers that its approval rate on healthy cards has fallen for reasons nobody can point at. The retry strategy that recovered an extra one per cent of failed subscriptions has cost two per cent of successful ones.

A timeout is not a decline, and treating it as one creates duplicates. This is the failure that produces the angriest customers. If a request goes out and no response comes back, the correct action is a reversal — a 0400 carrying the original message's trace number — followed by a fresh attempt. It is not a retry of the same request, because the first request may well have been approved by an issuer whose response was lost on the way home. Retry without reversal and the customer has two holds on their account for one purchase. This is why every terminal and gateway implementation must write a durable local record of the request, including the trace number, *before* the request goes on the wire: the case in which you most need to reverse is exactly the case in which nothing came back to tell you what to reverse.

Every attempt costs money even when it fails. Authorisation requests carry scheme fees, gateway fees and, in some acquiring arrangements, per-transaction charges regardless of outcome. At any volume the difference between ninety attempts and 1,484 is not a rounding error.

The retry logic that follows from all of this is short enough to state in full. Read the merchant advice code first if there is one, because it is an instruction and it overrides your own heuristics. If there is none, read the scheme's category. Never retry a Visa category 1 code, a Mastercard MAC 03 or 21, or any of R0, R1, R3. For data-quality declines, do not retry until the data has changed — a new expiry from the account updater, a corrected security code, a completed 3DS authentication. For temporary declines, retry on a schedule shaped by the customer's payday rather than by your batch window, and prefer the interval the issuer told you in MAC 24 to 30 if it gave you one. Cap total attempts well under the scheme threshold. Store the raw code, the scheme that produced it, the advice code and the timestamp for every attempt, and never collapse the outcome to a boolean, because the day you need to explain a fall in approval rates is the day you will need the codes.

Scheme differences in the code sets

The single most dangerous assumption in this area is that a two-character code means the same thing everywhere. It does not, and the collisions are not obscure.

Code	Visa	Mastercard	Discover
51	Not sufficient funds	Insufficient funds / over credit limit	Decline
70	PIN data required	Contact card issuer	not published in the same set
79	Already reversed (by Switch)	Life cycle	not published in the same set
82	Negative CAM, dCVV, iCVV, CVV, CAVV, dCVV2, TAVV or DTVV results	Policy	not published in the same set
83	Cannot verify PIN	Fraud/Security	Domain Restriction Controls Failure
85	No reason to decline	Not declined; valid for all zero amount transactions	No reason to decline

Code	Visa	Mastercard	Discover
1A	Additional customer authentication required	not used in this sense	Customer authentication required

Code 83 is the cleanest illustration: on Visa a PIN verification problem, on Mastercard a fraud decline, on Discover a token domain restriction failure. Three networks, one value, three unrelated events. A normalisation layer that maps `83` to a single internal meaning is wrong on at least two of them.

American Express does not use the two-character scheme at all in its own dialect. Its published set is three digits wide: `00` approved, `100` deny, `101` expired card or invalid expiration date, `109` invalid merchant, `110` invalid amount, `111` invalid account, `116` not sufficient funds, `117` invalid PIN, `122` invalid card security code, `130` additional customer identification required, `181` format error, `200` deny, pick up card, `912` issuer not available. An integration that assumes two characters will truncate every one of them.

The strong customer authentication soft decline deserves its own note because it is the newest common case and it is not handled correctly in a great deal of live code. Under the second Payment Services Directive an issuer may refuse a card-not-present transaction not because there is anything wrong with it but because the cardholder has not been authenticated. Visa signals this with `1A`, "additional customer authentication required". Mastercard signals it with `65`. American Express's `130`, "additional customer identification required", occupies the same position in its set. The correct response is neither a retry nor a failure message to the customer: it is to run the transaction through EMV 3DS, obtain the authentication data, and submit a new authorisation carrying it. A merchant that treats `1A` as a hard decline is turning a completable sale into a lost one, and a merchant that retries the identical message without authentication will receive `1A` again, indefinitely, while accumulating reattempt fees.

Note also that Mastercard's `65` is doing double duty — the ISO meaning is a withdrawal frequency limit and the European meaning in card-not-present contexts is an authentication requirement. Disambiguating it requires the context of the transaction rather than the code alone, which is a good final reminder that DE39 is two characters and two characters cannot carry a complete explanation of anything.

Codes the network writes rather than the bank

A final category worth separating out, because it changes what the right action is.

When an issuer's systems are unreachable, Visa's V.I.P. system can authorise on the issuer's behalf using parameters the issuer has lodged in advance. This is stand-in processing, and its fingerprints are all over the code set. Visa's own description of `91` is not "the issuer declined" but "issuer or switch inoperative and STIP not applicable or not available for this transaction; time-out when no stand-in". That is a statement about network availability, and it is the one decline where an immediate retry is genuinely the right move, because the condition it describes is measured in minutes. `N0`, "force STIP", is an issuer deliberately routing a transaction into stand-in rather than answering it itself. When stand-in approves, the approval code in DE38 was minted by the scheme and the issuer may not know about it until the advice message arrives, which is a small but real source of reconciliation surprise.

At the other end, `Z3`, "unable to go online; offline-declined", is written by the terminal after the chip demanded an online authorisation and the terminal could not get one. No bank saw the transaction at all. A chip that declines outright produces an application authentication cryptogram rather than an authorisation request cryptogram, and the terminal reports the refusal without any message leaving

the shop. In both cases the customer is told their card was declined while their bank has no record of anything having happened, which is exactly the conversation that ends with a customer insisting there is nothing wrong with their card and being entirely correct.

The practical instruction that closes the chapter is a small one. Log DE39 verbatim, as characters, alongside the scheme, the acquirer, DE38, DE44, the merchant advice code from DE48 where present, DE4, DE49, the MTI and the time. Do not normalise on the way in. Do not map to a boolean. The two characters are the most compressed piece of information in the entire payment system, and every layer that helpfully simplifies them destroys the only evidence you will have when somebody asks, six months later, why the approval rate moved.

29.98 Common wrong ideas

Wrong: A decline means the cardholder's bank refused the payment. **Right:** A transaction can be refused at five points — the chip, the terminal, the merchant's own fraud screening, the acquirer or its gateway, and the scheme standing in for the issuer — and the two characters do not say which, so "your bank declined this" is frequently untrue.

Wrong: `00` means yes and every other value means no. **Right:** Code 10 is a partial approval, 11 is an approval, 08 is an approval conditional on identification and 85 is the affirmative answer to a zero-amount verification, so software that treats anything other than `00` as a failure discards sales it had already won.

Wrong: An approval means the money has moved. **Right:** It means a bank has set some of its customer's money aside and promised to pay if you ask properly, and `00` in a reversal response says only that the reversal was accepted, which tells you nothing about the original.

Wrong: Soft and hard declines are categories defined by the standard. **Right:** Nowhere in ISO 8583 is there any such concept; the categories were invented afterwards by the schemes, published in rulebooks revised on an annual cycle, and they differ by scheme, differ by region and move.

Wrong: The schemes are gradually opening up the reasons behind declines. **Right:** What Visa's categories and Mastercard's merchant advice codes added is what the merchant should do next, not why the issuer said no; the reason itself remains available only to the cardholder, from their own bank, over an authenticated channel.

Wrong: A two-character code means the same thing on every network. **Right:** Code 83 is a PIN verification problem on Visa, a fraud decline on Mastercard and a token domain restriction failure on Discover, and American Express does not use two characters at all, so a normalisation layer that maps a code to one internal meaning is wrong on most of them.

Wrong: If no response comes back, send the request again. **Right:** The first request may well have been approved by an issuer whose response was lost on the way home, so the correct action is a reversal carrying the original trace number followed by a fresh attempt; retrying without reversing gives the customer two holds for one purchase.

Wrong: Retrying a decline costs nothing but a little effort. **Right:** The schemes charge for reattempts beyond published thresholds and from the very first reattempt on the never-retry category, issuers read the pattern as card testing and decline more of the merchant's healthy traffic, and every attempt carries scheme and gateway fees whatever its outcome.

Wrong: A stop-payment code is just another decline to work around. **Right:** R0, R1 and R3 are the cardholder speaking through their bank to stop this payment, all future payments to this merchant, or

all merchant-initiated payments, and a merchant that retries one is overriding an instruction its own customer gave to their own bank.

Wrong: Recording the outcome as approved or declined is enough. **Right:** Collapsing the response to a boolean, or storing the code without the scheme that produced it, destroys the only evidence available on the day somebody asks why the approval rate moved six months ago.

29.99 Chapter summary in 20 lines

1. A card authorisation reply carries its entire verdict in a single two-character field, DE39, and everything else in the message is corroboration.
2. `00` means approved, about sixty other values mean refused, and the gap between a payments operation that works and one that quietly bleeds money is almost entirely in what is done with those two characters.
3. The refusal is deliberately vague, because a specific reason is a free lesson for the next person who should not get in.
4. The arithmetic is the argument: a fully specific decline vocabulary turns an unstructured guess into a staged search, taking a bare card number to its expiry, its security code and its liveness in roughly a thousand probes.
5. Collapsing everything to `05`, do not honour, reduces the attacker to one bit per probe, and some declines — sanctions and money-laundering stops — cannot lawfully be explained to a merchant at all.
6. The disclosure is therefore directional rather than absent: the cardholder can ring their bank and be told exactly what happened, and the merchant cannot, because the merchant is statistically sometimes the attacker.
7. Refusals come in two flavours that look identical from the outside, temporary and permanent, which is precisely why blind retrying is so tempting and so expensive.
8. Ravinder's gym puts numbers on it: fifty-three failures treated identically generate 1,484 attempts and recover about £500, while treating them as five different problems generates about ninety attempts and recovers about £800.
9. Insufficient funds is a date problem that answers to a retry timed around payday, while an expired card, a lost card and a stolen card answer to nothing a retry can do.
10. Code 91 is the one entry in the list where "just try again" is the whole correct answer, because it describes a network outage measured in minutes.
11. Several values that are not `00` are nevertheless approvals — partial approval, VIP approval, honour with identification, no reason to decline — and treating them as declines gives away sales already won.
12. An approval is not a payment: it is a promise to pay if the merchant presents properly, and the authorisation identification response is populated only on approval, so deciding "approved" from its presence is the right answer for the wrong reason.
13. The 1993 revision replaced the two characters with a rationally structured three-digit action code, the card networks never adopted it, and as a result `10` and `100` are an approval and a decline distinguishable only by knowing which version is on the wire.
14. Soft and hard are not properties of the code but operational categories layered on afterwards, published in annually revised rulebooks and differing by scheme and by region.
15. Visa sorts every decline into four categories — never reattempt, cannot approve now, cannot approve on this data, and generic — with `05` sitting in the generic bucket the scheme is pressing issuers to shrink.

16. Mastercard went the other way, replacing the code with life cycle, policy and fraud values and putting the actual instruction in the merchant advice code, whose retry-after values tell the merchant how many days to wait.
17. Blind retrying damages a business in four ways: scheme reattempt fees, issuers scoring the traffic as card testing, duplicate holds when timeouts are mistaken for declines, and a per-attempt cost on every message.
18. A timeout demands a reversal carrying the original trace number, which is why that trace number must be written to durable local storage before the request ever goes on the wire.
19. The strong customer authentication soft decline is neither a retry nor a lost sale but an instruction to authenticate through EMV 3DS and submit a new authorisation carrying the result.
20. Log the response code verbatim as characters alongside the scheme, the acquirer, the authorisation code, the additional response data, the merchant advice code, the amount, the currency, the message type and the time, and never normalise on the way in, because every layer that helpfully simplifies those two characters destroys the evidence.

Sources: ISO 8583:1987 and ISO 8583:1993 data element and response code catalogues as tabulated in the ISO 8583 reference literature; Visa Developer published Request and Response Codes (action code, address verification result, CVV2 result and CAVV result tables); Visa BASE I / V.I.P. System technical specification descriptions of stand-in processing and offline-decline codes; Mastercard Switch Rules (Mastercard, 2016-2025) on DE 39 and DE 48 subelement 84 pairings; Mastercard Developers network response code and merchant advice code definitions; Worldpay Access published scheme response code tables for Visa, Mastercard, American Express and Discover, including Visa decline category assignments; Adyen raw acquirer response documentation and 3-D Secure soft-decline guidance; CardPointe and Payway merchant guidance on the Visa Excessive Reattempts rule; PayPal Braintree and TabaPay published merchant advice code tables.

Authorisation, Clearing, Settlement

30.0 What this chapter gives you

1. You will be able to explain why a customer's available balance falls in under two seconds while not one unit of currency has moved anywhere.
2. You will be able to separate authorisation, capture, clearing, settlement and merchant funding as five events with five different units of work, and say which of them actually move money.
3. You will be able to answer the question every support queue receives — why a refund takes days when the payment took seconds — without blaming the merchant, who in most cases did the work on day zero.
4. You will be able to explain to a driver at a motorway pump why £99.00 became £68.40 and why nobody has refunded them £30.60.
5. You will be able to say why a hotel that has cleared and reversed correctly can still leave a hold sitting on a customer's card, and why the customer's own bank is the only party that can lift it.
6. You will be able to code an estimated authorisation and its increments so that the indicators and the linking identifiers survive into clearing, which is the difference between a clean transaction and dispute liability.
7. You will be able to say why a reversal carries the original total in the amount field with the corrected figure in the replacement amount field, and what happens to the hold when those two are swapped.
8. You will be able to explain why storing a card for later use should be done with a zero-amount account verification rather than a one-pound test, and what the schemes now charge for the latter.
9. You will be able to model an authorisation as three joined records with three lifecycles rather than one row with a status column, and explain why a partly captured, partly reversed authorisation cannot fit in one row.
10. You will be able to recognise when you are in a single message system and stop applying dual-message vocabulary such as capture, void and batch cut-off, several of which do not exist there.

The plain version

On Amira's street the pocket money does not live in the children's pockets. It lives in a biscuit tin on Mrs Bello's kitchen shelf.

Mrs Bello has kept the tin for thirty years. Every child on the street has a page in a notebook beside it, and the page says how much of the tin is theirs. Amira's page says £62.30. There is no envelope in the tin with Amira's name on it. There is no separate £62.30. There is one tin with a great deal of money in it and a notebook that says who it belongs to. This matters later, so hold onto it.

On Thursday the ice cream van comes round. Amira wants a lolly. She does not have £62.30 in her hand; her money is in the tin. So the driver leans out of the window and shouts down the street to Mrs Bello: "Amira, up to six pounds, is that all right?"

Mrs Bello looks at the notebook, sees £62.30, and says yes. Then she does something very specific. She does not open the tin. She does not take out any money at all. She writes on a slip of paper "up to £6.00 promised to the ice cream van", clips it to Amira's page, and calls back "yes, that's fine".

Now look at what has and has not happened. The tin still contains exactly as much money as it did a minute ago. Not one coin has moved. But Amira's page now reads £62.30 with a £6.00 slip clipped to it, which means the most she can spend on anything else today is £56.30. Ask her how much money she has and the honest answer is two different numbers at once. She *has* £62.30. She can *spend* £56.30.

That is the whole trick, and it is the thing almost nobody knows. When your banking app says your money is gone, your money is usually not gone. A promise has been made about it. The promise is instant. The money is slow.

Why did the driver ask for six pounds when a lolly is four? Because he did not know yet. Amira might want a flake in it. She might buy one for her brother. He asked for enough to cover the likely worst case, because if he hands over the ice cream and then finds there is no money, the ice cream is not coming back. This is exactly why a hotel puts a hold on your card that is bigger than the room rate, and exactly why a petrol station holds an amount you never spent. Nobody knows what the final number is at the moment they have to commit.

That evening, at six o'clock, the driver goes back to the depot and writes out the day's notes. He does not write one note per ice cream and post them one by one. He writes the whole day at once, all forty-one sales, in one bundle, and puts the bundle in the post. Amira's line in the bundle says: **£4.20**.

The next morning the bundle arrives. Mrs Bello reads Amira's line, takes the slip off the page, opens the tin, and takes out £4.20. Only now has money moved. Amira's page reads £58.10, and she can spend £58.10, because the two numbers have finally become one number again.

Notice three separate events, at three completely different speeds.

The shout down the street took two seconds. That is **authorisation**. It reserves.

The bundle of notes took a day. That is **clearing**. It presents the real amount and says what actually happened.

And Mrs Bello opening the tin is **settlement**. That is the only step at which money actually moves.

There is one more layer, and it is the one that explains why nothing in this business is ever as fast as it looks. Mrs Bello does not walk £4.20 down to the ice cream depot. The depot has its own tin, kept by Mr Doyle, and on Friday afternoon Mrs Bello and Mr Doyle meet on the corner and square up the entire week. Amira's £4.20 has by then been added to four hundred other people's ice creams, and

subtracted from the eleven that got returned, and the whole street's week has become one figure and one handshake. Amira's £4.20 never made a journey of its own. It was absorbed into a number.

Now some things that go wrong, because they all follow from the same picture.

Amira changes her mind before the driver hands anything over. He shouts "cancel that one" and Mrs Bello unclips the slip immediately. Her page goes back to £62.30 within the minute. That is a **void**, or in the trade, a reversal, and it is the only fast way to undo a hold.

The driver forgets to shout. The slip stays clipped. There is no note in the bundle that evening, or the next, or the next. Amira's page still says she can only spend £56.30 and she cannot understand why. Mrs Bello has a private rule about this: if no note arrives within a week, she takes the slip off anyway. That is an **authorisation expiring**. It is worth being precise about whose rule that is. It is Mrs Bello's rule. Not the driver's, not the depot's. The person holding the money decides when to stop holding it, and everybody else can only ask.

Amira buys her lolly at five past six, just after the driver has sealed the bundle. Her line goes in tomorrow's bundle, which arrives the day after. That is a **batch cut-off**, and it is why something you bought on Friday evening appears on your statement on Tuesday.

And now the question everyone actually asks. Amira brings the lolly back unopened. The driver agrees to refund her. Why does it take four days when buying it took two seconds?

Because there is no shouting down the street in the other direction. The whole reason the payment felt instant is that a *promise* was made instantly, and a promise costs nothing. Giving money back is not a promise. It is the money itself, and the money has only ever had one speed. The driver writes a line in tomorrow's bundle saying "minus £4.20". The bundle goes in the post. It arrives. It gets read, checked and totalled against everything else. Then Mrs Bello puts £4.20 back in Amira's page.

Buying was fast because it was fake. Refunding is slow because it is real.

Here is the whole thing with real numbers. Tunde, Amira's father, fills up at a motorway service station on Thursday. The pump reads **£68.40**. He looks at his phone in the car park and it says **£99.00 pending**. He has not bought ninety-nine pounds of anything. The pump had to commit to a number before it knew what he would take, so it committed to a ceiling. On Friday the £99.00 disappears and **£68.40** appears in its place, and he assumes he has been refunded £30.60, which he has not, because the £99.00 was never taken.

The following Monday he checks into a hotel for two nights at £120 a night. Reception takes his card and authorises **£300.00**, being the £240 of room plus what the hotel genuinely expects him to spend on dinner. On Tuesday night he spends more than that, and at 21:10 the hotel quietly asks for another **£75.00** on top. He is now holding £375.00 of promises against a card he has used exactly once. On Wednesday morning the bill comes to **£341.20**. The hotel puts £341.20 into that night's bundle, and within twenty-four hours it is obliged to send a message releasing the £33.80 difference. If it forgets, the £33.80 sits on his card for a fortnight and he blames the hotel, which is fair, and phones his bank, which cannot help him, which is not obvious but is true.

Three events. Three clocks. One tin.

Where the plain version stops being true

The slip of paper is not a ring fence, and there is no tin. The picture of a hold as money set aside is comforting and wrong. An authorisation hold is not a transfer between two ledgers and it does not create a fund earmarked for the merchant. It is an adjustment to a derived figure — your *available* balance — held in the issuer's authorisation system, sitting on top of a posted balance that has not changed at all. The merchant has no claim on it, no security interest in it, and no ability to reach it. If your issuer becomes insolvent overnight, the hold protects the merchant not at all. And on a credit card there is no money involved at any point: the hold reduces your open-to-buy, which is a permission to borrow, not a sum you own. Two different things wear the same word.

Nothing in the picture obliges the person holding the money to let go of it. This is the correction that matters most, because it is the one that misdirects a million customer complaints a year. The card schemes publish rules about when a *merchant* must clear or reverse an authorisation. They do not, in the general case, publish rules obliging an *issuer* to drop a hold at a given hour. Those are different obligations pointing in different directions. So the merchant can do everything correctly — clear on the day, reverse the excess within twenty-four hours — and the hold can still sit on your account for days, because the issuer's matching logic did not connect the two records, or because its release policy runs on a timer. When a hotel tells you "we've released it, it's with your bank now", that is very often the literal truth and it helps you not at all.

"Three events" describes one architecture, not all of them. The separation of authorisation from clearing is what the industry calls a dual message system, and it is the architecture of most credit card traffic. There is another architecture, the single message system, in which one message both asks permission and moves the money, and there is no separate presentment at all. ATM withdrawals work that way. So does a great deal of debit traffic in various markets. Once you are in single-message territory, "capture", "void" and "batch cut-off" are not merely different — several of them do not exist as concepts, and a reversal is a different animal with a different deadline. Any statement in this chapter about three timescales is a statement about dual message processing.

Settlement between the banks is not the merchant being paid. These are two distinct events, days apart, and the industry uses the same word for both, which is a permanent source of confusion in merchant support queues. The scheme calculates a net position and moves funds between the issuer's and the acquirer's settlement banks. Then, separately, under a commercial contract that the scheme is not party to, the acquirer pays the merchant — net of interchange, scheme fees and its own margin, possibly minus a rolling reserve, on a schedule that might be the next working day or might be five days, and which travels over Bacs or Faster Payments like any other bank payment. A merchant asking "has it settled?" and an acquirer answering "yes" are frequently talking about different days.

The technical version

Three events, three clocks

The three events differ not only in when they happen but in what their unit of work is. Authorisation is per transaction. Clearing is per file. Settlement is per institution. Losing track of which unit you are in is the root of most reconciliation failures.

Event	Typical vehicle	Latency	Unit of work	Moves money
Authorisation	ISO 8583 `0100` / `0110`	300 ms to 2 s	One transaction	No

Event	Typical vehicle	Latency	Unit of work	Moves money
Capture	Terminal batch close, acquirer capture file	Minutes to 24 hours	One merchant's day	No
Clearing	Visa BASE II `TC05`, Mastercard IPM `1240`	1 to 3 days	One institution's file	No
Settlement	Net position, funds transfer	1 to 3 business days	One institution's net position	Yes
Merchant funding	Bacs or Faster Payments credit	T+1 to T+5 by contract	One merchant's contract	Yes

Only the last two rows move money, and only the last row moves it to the party the customer thinks they paid.

Authorisation: what the message does and does not do

An authorisation request is an ISO 8583 `0100`, answered by a `0110`. Its four-digit message type indicator decomposes as version, class, function, origin: `0` for the 1987 edition, `1` for authorisation class, `0` for request, `0` for acquirer origin. In single message systems the equivalent is a `0200` financial request answered by a `0210`, and the distinction is exactly the one drawn above — a `0200` is understood to post.

The fields that carry the commercial substance of the request are the ones catalogued earlier in this volume. DE2 carries the primary account number. DE3 carries the processing code, six digits of transaction type plus from-account plus to-account. DE4 carries the amount in minor units of the currency named in DE49, itself an ISO 4217 numeric code, `826` for sterling, `978` for euro, `840` for US dollars. DE7 carries transmission date and time, DE11 the six-digit system trace audit number, DE12 and DE13 the terminal's own local time and date. DE37 carries the twelve-character retrieval reference number and DE41 the eight-character terminal identifier. DE55 carries the EMV data. On the way back, DE38 carries the six-character authorisation identification response, DE39 the two-character response code.

What the `0110` does is create an obligation with an expiry. It does not create a payment instruction. Nothing in the authorisation pair causes a single unit of currency to change hands, and an acquirer that never follows it with a presentment will never be paid, however emphatically the `0110` said approved.

ISO 8583 has an explicit field for the lifetime of that obligation. DE57, the authorisation life cycle, is three characters: position one is a time code, where `1` denotes calendar days, `2` hours and `3` minutes, and positions two and three carry an interval of `01` to `99`. So `102` is two calendar days and `224` is twenty-four hours. The standard's action code list likewise carries `1033` (authorization lifecycle unacceptable) and `1034` (authorization lifecycle has expired), which tells you that the drafters expected disagreement about lifetimes to be a normal, coded outcome rather than an operational incident. In practice the schemes largely govern this through their own rulebooks rather than through DE57, but the field exists and some domestic switches use it.

What the issuer actually does to the balance

An issuer holds at least two figures against an account and returns different ones to different channels.

The *posted* or *ledger* balance is the sum of transactions that have actually been applied to the account. It changes when clearing arrives, not when authorisation happens.

The *available* balance is a derived figure: posted balance, plus or minus any credit line, minus the total of outstanding authorisation holds, minus any uncleared-funds or fraud-related restraints. This is the number the authorisation system tests a `0100` against, and it is the number your app shows you.

The pending items you see listed in a banking app are outstanding authorisations, not transactions. They can change amount. They can vanish without ever posting. They can post at an amount that never appeared in the pending list at all. Support scripts that tell customers a pending item "will be taken tomorrow" are asserting something the issuer does not actually know.

When the presentment eventually arrives, the issuer must decide whether it corresponds to an existing hold. If it matches, the hold is released as the posting is applied and the customer sees one clean line. If the matching fails, the issuer applies the posting *and* leaves the hold in place until it ages out, and the customer sees the same amount twice. That is the mechanism behind the overwhelming majority of "I've been charged twice" calls, and it is why the scheme processing rules are so preoccupied with identifiers.

Estimated and incremental authorisation

Where the final amount is genuinely unknown at the point of commitment, Visa's rules permit an estimated authorisation followed by any number of incremental ones. The important word is genuinely: Visa's own merchant guidance states that an estimated authorisation "must be a genuine estimate and must not be an arbitrary amount", must not include incidental amounts such as tips or a damage buffer, and must not be used to check the status of a credential — that is what account verification exists for.

The coding requirements are specific, and they are the difference between a clean transaction and a hold the issuer cannot match. In Visa's message structure the fields are private-use subelements:

Purpose	Visa field	Value
Flag an estimated authorisation	Field 60.10, Additional Authorization Indicator	`2` estimated amount, or `3` estimated amount and terminal accepts partial approvals
Flag an incremental authorisation	Field 63.3, Message Reason Code	`3900`
Flag an incremental authorisation (US alternative)	Field 62.1, Authorization Characteristics Indicator	`I`
Link every message to the original	Field 62.2, Transaction Identifier	The TID Visa returned on the original approval
Link every message to the original	DE37, Retrieval Reference Number	The value from the original request

The amount semantics differ between message types and this trips people up constantly. In an incremental authorisation, the amount field carries the *additional* amount being requested, not the new total. In an authorisation reversal, the amount field carries the *original* total authorised — the sum of the estimate and all increments — and the corrected figure goes in the Replacement Amount field.

Failure to code these correctly is not merely untidy. Visa states that acquirers and merchants may carry dispute liability under Dispute Condition 11.3, No Authorization / Late Presentment, where there is no estimated indicator on the first message, no incremental indicator on subsequent messages, a mismatched Transaction Identifier, or processing outside the permitted timeframe.

Those timeframes are, per Visa's 2024 merchant guidance, the maximum time from a valid estimated authorisation to processing:

Segment	Maximum
Lodging, vehicle rental, cruise line	30 days from estimated authorisation approval
Rental merchant categories	10 days
Cardholder-initiated card-absent transactions	10 days
Card-present transactions	5 days

Two cautions on that table. First, Visa notes country-specific approval response validity timeframes exist and the rulebook governs. Second, and instructively, the 2023 edition of the same Visa document published a materially different set: 31 days for lodging, vehicle rental and cruise lines, 7 days for other rental categories and other card-absent transactions, 3 or 7 days for commuter transport, and same-day for other card-present transactions. Both documents are Visa's. If you are hard-coding validity windows, hard-code them against the rulebook edition in force and version them, because they move.

Incremental authorisations do not extend the window. A stay or rental that outlives the validity period must be closed within it and reauthorised, and the new authorisation gets a fresh window of the same length.

Fuel, and why the number is never the number

Automated fuel dispensers do not use the estimated and incremental framework. They use an older construct, the initial authorisation, in which the terminal sends a fixed amount before the final figure is known, the amount is capped, no incremental is permitted, and the pump must stop if the fill reaches the authorised ceiling. Visa's guidance notes that from April 2025 only AFDs may use initial authorisations, with laundries, quick copy, car wash, video rental and vending encouraged to migrate to estimates and increments.

The best known variant is the one-dollar status check, in which the terminal authorises a nominal amount and the issuer is expected to place a hold reflecting the segment ceiling rather than the nominal amount. Visa's own business news of that change, effective 17 October 2020, sets the US AFD status check authorisation limits at **USD 125 for non-Fleet cards and USD 350 for Fleet cards**, applying only where the transaction is both chip-on-chip and partial-authorisation participating, and leaving them at **USD 100 and USD 150** otherwise. Issuers remain liable up to those limits and may dispute the excess under Reason Code 11.3.

Those ceilings have since moved again, and the figure to know now is USD 175. It circulates very widely in acquirer-facing and merchant-facing material as "the" AFD limit, and although it appears nowhere in the 2020 notice it is not folklore. Visa's own fuel segment update, presented by a Visa senior business leader to the National Petroleum Energy Credit Association in May 2025, gives the maximum US status check amount as USD 175 for non-Fleet cards and USD 1,000 for Fleet cards, and warns in the same deck that issuers who hold the nominal one dollar rather than that maximum are being exploited by fraudsters, which is the whole mechanism stated in reverse. The general point survives the revision: these ceilings are rewritten every few years, so version them against the edition of the rules in force rather than against whatever number the internet is quoting this season.

The same notice describes the mechanism that actually shortens fuel holds, and it is worth knowing because it is the only widely deployed example of a message whose sole purpose is to shrink a hold in real time. Visa asks issuers to activate their BINs for enhanced AFD and real-time clearing confirmation advices, which carry the final amount ahead of settlement, and notes that settlement "can take up

to two days following completion of the AFD transaction". Issuers that apply the advice immediately release the hold in minutes. Issuers that do not, wait for clearing, and their cardholders are the ones writing to the newspapers.

Partial approval

If the account cannot cover the requested amount, an issuer that supports partial approval may approve a smaller one instead of declining. Under the 1987 response code list DE39 carries `10`, approved for partial amount; the third-edition action code list carries `0002` with the same meaning. The approved amount is returned in the amount field, and the terminal is expected to prompt for the balance by another tender.

The terminal has to have declared that it can cope with this, which is what the value `3` in Visa's Additional Authorization Indicator does: estimated amount *and* terminal accepts partial authorisation responses. A terminal that has not declared the capability and receives a partial approval anyway will typically treat it as a full approval for the requested amount, which is a genuine and repeatable way to give away goods.

Reversals, voids and cancellations

A reversal is the only fast instrument in the entire chapter. It is an ISO 8583 `0400` reversal request, or a `0420` reversal advice with its `0430` response where the recipient has no discretion to refuse.

Two fields carry the work. DE90, original data elements, is a forty-two digit block that identifies the message being reversed: original MTI (4), original STAN (6), original transmission date and time (10), original acquiring institution identification code (11) and original forwarding institution identification code (11). DE95, replacement amounts, forty-two characters, carries the corrected amounts where the reversal is partial rather than full; Visa's equivalent in its own message structure is the Replacement Amount field. Sources differ on whether DE95 is specified as numeric or alphanumeric, and scheme implementations vary in how they subdivide it, so treat the subfield layout as scheme-specific rather than universal.

The industry word "void" is a merchant-facing abstraction, not a message. Depending on where the transaction has got to, a void is implemented either as a reversal of an authorisation that has not yet been captured, or as the removal of a record from a batch that has not yet been submitted. Once the batch has gone, there is no void; there is only a refund, with everything that implies about timing.

Visa's reversal obligations are expressed in hours, not days:

- Where the transaction will not be completed at all, the entire authorised amount must be reversed within 24 hours of the earlier of the merchant becoming aware of that fact or the end of the authorisation validity period.
- Where the transaction completes and the sum of the estimate and any increments exceeds the final amount, the difference must be reversed within 24 hours of completion.

Again, the 2023 edition of the same guidance expressed the second obligation with tolerances — a 15 per cent threshold for lodging, vehicle rental and cruise lines, 20 per cent including tips for taxis — before the 2024 edition stated it flatly. Read the rulebook edition you are actually operating under.

Capture and the batch cut-off

Between the `0110` and the clearing file sits capture, which is where a merchant declares that the authorisation should become money. In an attended retail environment the terminal accumulates approved transactions locally and closes its batch, either on a schedule or on an operator action. In an

integrated or gateway environment the merchant sends an explicit capture instruction, which is why an e-commerce platform can authorise at checkout and capture at despatch three days later.

On the acquirer side, the file that carries this into Visa's clearing system is the **TC 33.A Capture File**, whose records are laid out in *BASE II Clearing: Interchange Formats, TC 01 to TC 49*. It carries purchase and refund transactions that already have approved authorisations, decomposed into transaction code records: `CP01` transaction, additional, billing and shipping, merchant, instalment, gateway and supplemental data; `CP02` EMV data; `CP03` lodging summary; `CP04` Level II and Level III purchasing data; `CP05` air passenger itinerary; `CP10` car rental; and so on. The Edit Package, Visa's own validation software that acquirers install, wraps the file with a `TC90` file header, `TC91` batch record and `TC92` file trailer, and validates that the `TC05` (Sales Draft) and `TC06` (Credit) transactions built from the capture file are correct before they go to Visa for clearing and settlement.

The cut-off is a property of every one of these hops, and they are not aligned. A terminal that closes its batch at 23:00 local, an acquirer that assembles its submission at 01:00, and a scheme clearing window that closes at a fixed time in a different time zone together produce the familiar phenomenon whereby a Friday evening purchase is presented on Monday and funded on Wednesday. None of those systems is slow. They are each waiting for the next one's door to open.

Clearing

Clearing is the exchange of transaction detail between acquirer and issuer, the assessment of interchange and fees, and the production of the figures that settlement will act on. It is a batch process, it is file-shaped, and it is where the transaction acquires the identifiers that disputes will later be filed under.

Visa clears through **BASE II**, whose record vocabulary is the transaction code series already described: `TC05` for a sales draft, `TC06` for a credit. Mastercard clears through the **Global Clearing Management System**, using the **Integrated Product Messages** format. In IPM the first presentment is message type `1240` with DE24 function code `200`; a financial detail addendum is `1644` with function code `696`; the first chargeback is `1442`. DE25 carries the message reason code, and Mastercard's switch rules use reason code `1404`, "previously approved authorization — partial amount, final clearing", to signal that a presentment is the last one against a given authorisation and that the authorisation is now closed, after which no further clearing messages may be submitted against it. Mastercard's clearing dates are expressed as the Central Site Business Date, which is the date the message cleared into GCMS and not the date anything happened in a shop.

Presentment is not optional and it is not open-ended. Mastercard's European rules require an ATM presentment within seven calendar days of the transaction date, with late presentment exposed to chargeback under message reason codes `4842`, `4880` for Maestro ATM, or `4811` for stale transactions, and transactions presented between forty-six calendar days and one year attracting a fee that is passed in full to the issuer. The same shape of rule, with different windows, applies across the card products.

Two asymmetries in clearing deserve stating plainly, because both break the tidy mental model.

A clearing record can exist with no authorisation behind it. Offline-approved transactions, forced posts and certain fallback flows all produce presentments the issuer never saw a `0100` for. The issuer must post them and argue afterwards.

And a clearing record's amount is not obliged to equal the authorised amount. Tolerances vary by segment, and where the clearing amount exceeds what the tolerance allows, the issuer's remedy is a dispute rather than a rejection at the door.

Settlement

Settlement is the discharge of the obligations that clearing recorded, and it is the only step in this chapter at which value moves between institutions.

Mastercard describes its own architecture in three named systems: the Authorization Platform, the Global Clearing Management System for clearing, and the **Settlement Account Management** system for settlement, which calculates the net position of each acquirer and issuer arising from clearing and performs two functions, sending advisements and transferring funds. Visa's equivalent function is performed by the VisaNet Settlement Service, fed by the figures BASE II produces.

Three properties follow, and each of them surprises somebody.

Settlement is *net*, not gross. An issuer that owes on ten million transactions and is owed on two million settles one figure. This is the netting described in Volume I, applied at institutional scale, and it is why a card scheme's daily settlement flows are a small fraction of its daily transaction value.

The scheme is not a bank. It computes positions and instructs; the actual transfer happens between nominated settlement banks over ordinary payment infrastructure, in a settlement currency chosen per region and per member. This is the structural reason a scheme outage stops new authorisations but does not destroy value that has already cleared.

And interchange is not invoiced. It is applied as an adjustment inside the clearing figures, so by the time a settlement position exists it is already net of the interchange the acquirer owes. There is no separate interchange bill to pay or dispute.

Then, entirely separately, the acquirer pays the merchant. That payment is governed by the merchant agreement, not the scheme rulebook. It is net of the merchant service charge, it may be net of a rolling reserve, and it travels over whatever domestic rail the acquirer uses — in the United Kingdom, Bacs or Faster Payments, with the timings Volume IV sets out. T+1 and T+3 are both common; new or higher-risk merchants are routinely funded more slowly. A merchant seeing money on Wednesday for a Monday sale is seeing the end of a chain in which the scheme settlement happened on Tuesday and had nothing to do with them.

Why the refund takes days

Put the two directions side by side and the asymmetry is obvious.

A purchase gets a real-time promise and a slow money movement. The customer experiences the promise. Their available balance drops in under two seconds and they conclude, reasonably, that the payment system is fast.

A refund has historically had no promise at all. The refund was a clearing-side credit — a `TC06` in BASE II terms — which meant it travelled at the speed of files: the merchant's batch, the acquirer's submission, the scheme's clearing cycle, the issuer's posting run. Nothing was reserved, because there is nothing to reserve; the issuer is not going to promise you money it has not received. Worldpay's own documentation puts the historical position exactly: issuers "accepted and honoured refund transactions when they received them later in the daily clearing files sent to them by processors", and "typically, this would take 24 hours for a refund transaction to be registered in cardholders' accounts", before the rest of the posting cycle ran.

The customer therefore experiences the purchase as instant and the refund as a five-day wait, and concludes that the merchant is stalling. In most cases the merchant did the work on day zero.

Both schemes have addressed this, and it is worth being precise about what they addressed, because it is a common and expensive misreading. Visa Purchase Return Authorization and Mastercard's equivalent return authorisation mandate require the merchant to obtain an *online authorisation for the refund*. Verifi, a Visa company, answered the obvious question about it in terms that leave no room: the mandate "is intended to increase consumer visibility to refunds in process by making pending refunds visible via online and mobile banking channels", and the time between refund issuance and funds appearing "can vary based on the processing timelines for each party".

In other words, the refund authorisation creates the missing promise. It does not create a faster money movement. Your bank can now show you a pending credit within seconds. The money still arrives when clearing and settlement bring it.

The compliance timeline, as of writing, is worth recording because the deadlines differ by region and by scheme:

Requirement	Region	Date
Refund without valid authorisation exposed to issuer dispute under reason code 11.3	US, Visa	From April 2020
Return authorisation without matching settlement record included in Visa's Misuse of Authorization programme	US, Visa	From July 2020
Online authorisation required for all refunds, credit vouchers and purchase returns	Europe: UK and Ireland, Visa and Mastercard	From April 2025
Mastercard non-compliance fees commence	UK and Ireland	1 May 2025
Visa non-compliance fees commence, at USD 0.05 per unauthorised refund	UK and Ireland	19 October 2025

Two operational consequences follow that catch integrations out. A refund authorisation can be *declined*, which was previously impossible, so a point-of-sale application must be able to render a declined refund without falling over. And an approved online refund typically returns a six-digit authorisation code, while an offline-accepted refund may return a five-digit one, which is a useful diagnostic when reconciling a merchant estate mid-migration. Visa publishes suggested merchant actions per decline code for returns, of which the useful ones are 14 invalid account number, meaning request an alternative card; 54 card expired, meaning ask whether a replacement exists; and 55 invalid PIN, meaning retry PIN normally.

The integrity fees, and what they are really for

The schemes charge for authorisations that never become money. The framing is data integrity, and it is honest framing: an unmatched authorisation is a hold on a customer's account for which no corresponding value will ever exist.

Visa's **Misuse of Authorization System Fee** applies in the US region to approved and partially approved authorisations that cannot be matched to a clearing transaction or an authorisation reversal. It was long assessed at USD 0.09 and was raised to **USD 0.15** with effect from January 2025 according to acquirer pass-through bulletins. Visa lists the two most common causes as failure to process estimated authorisations, incrementals and reversals to the technical requirements, and use of one-dollar

test transactions where the Account Verification Service should have been used. Visa also operates a separate unmatched clearing fee for the mirror-image failure. The same January 2025 schedule increased the Visa BASE II transmission fee from USD 0.0018 to USD 0.0025 per transaction, a figure worth knowing chiefly because it demonstrates how thin the per-item economics of clearing actually are.

Mastercard's **Processing Integrity Fee** turns on authorisation type, and the type is declared by the merchant:

Authorisation type	Requirement
Pre-authorisation	Fully reversed or cleared within 30 calendar days of the authorisation date
Final authorisation	Fully reversed or cleared within 7 calendar days; clearing amount must equal the authorised amount; clearing currency must match the authorised currency
Undefined authorisation	Cleared within 7 calendar days; Mastercard has been progressively restricting this category

The fee rates are region-specific and revised regularly. Fiserv's published Canadian pass-through schedule, as one concrete example of the shape of them, lists all three processing integrity non-compliance categories at 0.452 per cent with a USD 0.113 minimum. Do not carry a rate across regions; look it up in the schedule that applies to your acquiring entity.

The design lesson embedded in the final-authorisation rule is the sharpest one in this chapter. Mastercard will charge you if the amount you clear is not the amount you authorised, *when you told it the amount was final*. The rulebook is not asking you to be accurate. It is asking you to be honest about whether you know.

Account verification, which is not an authorisation

The correct way to check that a card exists and is in good standing is a zero-amount account verification: a `0100` with the amount field at zero, which Visa exposes as its Account Verification Service. It returns an approval and, on most implementations, address and security code verification results, without creating a hold.

The incorrect way, still widely deployed, is a small-value authorisation — the one-dollar or one-pound test — which creates a real hold on a real customer's real available balance, and which the schemes now charge for when it is never cleared. If you are storing a card for later use, verify at zero.

Design rules that follow

Model the three events as three separate records with three separate lifecycles, joined by identifiers, and never as one row with a status column. An authorisation that is partially captured and partially reversed has produced four facts, and a single row cannot hold four facts.

Store the linking identifiers on every message. For an estimated-and-incremental flow this means the scheme transaction identifier, the retrieval reference number and the system trace audit number carried forward from the original approval into every increment, every reversal and the clearing record. This is not good practice; per Visa's rules it is the difference between a clean transaction and dispute liability under condition 11.3.

Reverse promptly and reverse for the right amount. The obligation is measured in hours from the moment you knew, not days from the moment you got round to it, and the reversal message must

carry the original total in the amount field with the corrected figure in the replacement amount field. Getting those two the wrong way round produces a hold that grows rather than shrinks.

Declare your authorisation type honestly. If you know the final amount, say final and then clear that exact amount in that exact currency. If you do not know, say estimated and use increments. The undefined category exists because implementations were vague, and it is being closed.

Never tell a customer that a pending item will be taken. Tell them it is a reservation, that it may change or disappear, and that the amount that will finally post is the amount on the receipt. Every one of those statements is true, and the sentence "it'll come off tomorrow" is not.

And when a customer asks why the refund is slow, the honest answer is the one this chapter began with. The payment was not fast. The promise was fast. The money has always taken days, in both directions, and the refund is simply the first time anybody notices.

30.98 Common wrong ideas

Wrong: A hold sets your money aside for the merchant. **Right:** It adjusts a derived figure, the available balance, inside the issuer's authorisation system; it creates no fund, gives the merchant no claim and no security interest, and on a credit card no money is involved at any point, only a reduction in a permission to borrow.

Wrong: The merchant is holding the money and the merchant can release it. **Right:** The scheme rules say when a merchant must clear or reverse; they do not in the general case oblige an issuer to drop a hold at a given hour, so "we've released it, it's with your bank now" is very often the literal truth and helps the customer not at all.

Wrong: Every card transaction has three separate events. **Right:** That describes dual message processing; in single message systems one message both asks and moves, there is no separate presentment, and capture, void and batch cut-off are not merely different but absent as concepts.

Wrong: Settlement means the merchant has been paid. **Right:** Scheme settlement moves a net position between the issuer's and the acquirer's settlement banks, and the acquirer then pays the merchant days later under a commercial contract the scheme is not party to, net of charges and possibly a rolling reserve.

Wrong: A pending item will be taken tomorrow at the amount shown. **Right:** Pending items are outstanding authorisations, not transactions: they can change amount, vanish without ever posting, or post at a figure that never appeared in the pending list.

Wrong: "I've been charged twice" means the merchant sent the transaction twice. **Right:** The overwhelmingly common mechanism is that the presentment failed to match the hold, so the issuer applied the posting and left the hold in place until it aged out.

Wrong: The refund authorisation mandates have made refunds faster. **Right:** They create the promise that refunds never had, so a pending credit becomes visible within seconds; the money still arrives at the speed of clearing and settlement, which has not changed.

Wrong: An incremental authorisation carries the new total. **Right:** It carries only the additional amount; it is the reversal that carries the original total, with the corrected figure in the replacement amount field, and getting those the wrong way round produces a hold that grows instead of shrinking.

Wrong: An estimated authorisation can be whatever figure is convenient. **Right:** It must be a genuine estimate, must not include incidentals such as tips or a damage buffer, and must not be used to check the status of a credential.

Wrong: A one-pound test is a harmless way to check that a card is real. **Right:** It creates a real hold on a real customer's real available balance and is chargeable when it never clears; the correct instrument is a zero-amount account verification.

30.99 Chapter summary in 20 lines

1. A card payment feels instant because a promise is made instantly, and a promise costs nothing to make.
2. Authorisation reserves, clearing presents the real amount, and settlement is the only step at which value moves between institutions.
3. A hold is not money set aside but an adjustment to a derived available balance sitting on top of a posted balance that has not changed.
4. That is why a cardholder honestly has two numbers at once: what they have, and what they can spend.
5. Merchants ask for more than they need whenever the final amount is unknown at the moment of commitment, which is why a hotel holds more than the room rate and a pump holds more than the fill.
6. Capture is the merchant declaring that an approval should become money, and it sits between the authorisation response and the clearing file.
7. Clearing is file-shaped and batch-driven, carried by Visa's BASE II transaction codes and Mastercard's IPM messages, and it is where a transaction acquires the identifiers that disputes will later be filed under.
8. Settlement is net rather than gross, is computed by the scheme and executed between nominated settlement banks, and already has interchange applied inside it, so there is no interchange bill to pay.
9. The acquirer then pays the merchant separately, under a merchant agreement rather than the scheme rulebook, on a schedule that may be anywhere from the next working day to five days later.
10. Cut-offs exist at every hop and are not aligned, which is why a Friday evening purchase is presented on Monday and funded on Wednesday, and why none of the systems involved is actually slow.
11. A reversal is the only fast instrument in the whole chapter, and the obligation is measured in hours from the moment the merchant knew, not days from the moment it got round to it.
12. Where the final amount is genuinely unknown, an estimated authorisation followed by increments is the sanctioned pattern, on condition that the estimate is genuine and every message carries the linking identifiers forward.
13. Failing to code those indicators and identifiers is not untidiness; it moves dispute liability onto the acquirer and the merchant.
14. Fuel dispensers use an older construct with a segment ceiling instead, and the published ceilings are rewritten every few years, so they must be versioned against the rulebook edition in force rather than against whatever figure is circulating.
15. Partial approval lets an issuer approve a smaller amount than was asked, and a terminal that never declared the capability will treat the smaller approval as a full one, which is a repeatable way to give goods away.
16. Refunds historically had no promise attached at all, because the issuer will not promise money it has not received, so they travelled at the speed of files.

17. Refund authorisation mandates supply the missing promise and make a pending credit visible immediately; they do not make the money arrive any sooner, and a refund can now be declined, which point-of-sale software must handle.
18. The schemes charge for authorisations that never become money, because an unmatched authorisation is a hold on a customer's account for which no value will ever exist.
19. Mastercard's final-authorisation rule carries the sharpest lesson in the chapter: the rulebook is not asking a merchant to be accurate, it is asking it to be honest about whether it knows.
20. So the honest answer to a customer asking why the refund is slow is that the payment was never fast — the promise was fast, the money has always taken days in both directions, and the refund is simply the first time anybody notices.

Sources: Visa, Estimated and Incremental Authorization and Reversal Processing Requirements for Visa Merchants (2024 and 2023 editions); Visa Business News, U.S. Automated Fuel Dispenser Authorization Limits Will Be Increased, effective 17 October 2020; Visa Merchant Resource Library; Verifi (a Visa company), Visa Purchase Return Authorization FAQ; Visa Acceptance Solutions Payments Acquirer Implementation Guide, TC 33.A Capture File and Edit Package (BASE II Clearing: Interchange Formats, TC 01 to TC 49); Mastercard, Switching explained (Authorization Platform, GCMS, Settlement Account Management); Mastercard Switch Rules manual; Mastercard Chargeback Guide, clearing and processing cycles; Mastercard IPM Clearing Formats message type and function code tables; ISO 8583 third-edition Annex D code listings as circulated by Accredited Standards Committee X9 (action codes, authorization life cycle codes); ISO 8583 data element definitions for DE90 and DE95; Worldpay Total Hospitality documentation, Online Refund and the online authorisation of refunds mandate; Fiserv Merchant Services published pass-through fee schedule; acquirer pass-through bulletins for January 2025 Visa and Mastercard fee changes.

Where the Money Goes

31.0 What this chapter gives you

1. You will be able to say who pays interchange and who receives it, and explain why the merchant is neither of those parties despite being the one out of pocket.
2. You will be able to break a merchant service charge into interchange, scheme fees and acquirer net revenue, and name the single component that is negotiable.
3. You will be able to explain why the issuer's slice on a £480 corporate card ticket is over a thousand times the slice on a £4.20 debit card lolly when the ticket costs only 114 times as much.
4. You will be able to list the inputs a scheme's rating engine uses to derive interchange, and say why a transaction's rate is not knowable with certainty at the point of sale.
5. You will be able to state exactly how far the UK interchange caps reach, and name the exclusions — commercial cards, cash withdrawals, three-party schemes and any cross-border pair — that put a great deal of traffic outside them.
6. You will be able to explain why two identical £100 online sales can generate 30 pence and £1.50 of interchange, and why the merchant can neither detect, price for, nor decline the difference.
7. You will be able to name the four pricing models the regulator recognises and explain why over 95 per cent of merchants are on the one that shows them nothing.
8. You will be able to show, with arithmetic, why a merchant taking £2,000 a month through a single terminal can be paying 2.75 per cent before a single transaction has been rated.
9. You will be able to cite the article of the Interchange Fee Regulation that entitles a merchant to unblended pricing, and the article that entitles it to transaction-level interchange data.
10. You will be able to explain why a coffee shop's effective rate can exceed an airline's, and why neither merchant controls the thing that actually decides it.

The plain version

The ice cream driver from the last chapter sold Amira a lolly for £4.20. On Friday afternoon, when Mr Doyle at the depot squares up the week and pays the drivers, that lolly appears on the driver's payslip as **£4.05**.

The driver did not agree to a discount. Amira did not pay less. Fifteen pence went somewhere, and the driver has never met the person who took it.

Here is where it went, and the answer is stranger than anyone expects.

Start with Mrs Bello, who keeps the biscuit tin and the notebook. When the driver shouted down the street on Thursday, it was Mrs Bello who had to stop what she was doing, find Amira's page, check the number, decide whether to say yes, write the slip and clip it on. It was Mrs Bello who took the risk

that Amira's page was wrong. It is Mrs Bello who has to argue with Amira in three weeks' time when Amira insists she never bought the lolly at all. And it is Mrs Bello who has to give Amira her money back out of the tin if the lolly turns out to have been off, whether or not the driver ever gives it back to Mrs Bello.

Mrs Bello does all that work for the driver's benefit. So Mrs Bello keeps a slice.

That slice is the thing the industry calls **interchange**, and it is the single most misunderstood number in the whole of payments, because it runs the wrong way. Ask anyone who pays it and they will say the shop pays it to the card company. Both halves of that are wrong. The shop's bank pays it, and it goes to *your* bank. It is a payment from the seller's side of the street to the buyer's side of the street, and the seller has no relationship with the person receiving it and no ability to argue about the price.

Now the second slice. Somebody printed the notebooks. Somebody wrote the rules that say Mrs Bello must answer within two seconds, that the driver may not add a surcharge, that the Friday meeting happens at four o'clock on the corner and not whenever anyone feels like it. Somebody decides what happens when Amira and the driver disagree. That is Miss Vaughan, who owns the rulebook. She was never in the room when the lolly was sold and she never touches the money. She takes a small slice from Mrs Bello and a small slice from Mr Doyle, every week, for keeping the street running.

That slice is **scheme fees**, and Miss Vaughan is Visa or Mastercard.

And now the third slice, which is the only one that goes to somebody the driver has actually met. Mr Doyle at the depot collected the bundle, checked it, chased the missing lines, argued with Mrs Bello about the disputed lolly, and — this is the important bit — paid the driver on Monday even though the money from Mrs Bello did not arrive until Wednesday. If the driver takes a hundred pounds of orders and then vanishes, Mr Doyle is the one left holding it. For that, Mr Doyle keeps a slice too.

That slice is the **acquirer's margin**.

Add all three together and you get the number the driver actually cares about, which is the gap between £4.20 and £4.05. In the trade the whole gap has a name: the **merchant service charge**, or MSC. It is not one fee. It is three fees wearing one coat, and only one of the three is negotiable with the person who sends the invoice.

Now the real numbers, and this is where it gets properly interesting.

The lolly. Amira paid £4.20 with an ordinary British debit card, tapped on the driver's reader. The rate Mrs Bello is allowed to keep is 0.20 per cent. Twenty pence in every hundred pounds. On £4.20 that is **0.84 pence**. Less than a penny. Mrs Bello did all that work — the check, the promise, the argument in three weeks, the refund risk — for less than one penny.

The aeroplane ticket. On the same afternoon, in an office two miles away, someone books a £480 flight on their company's corporate card, typing the number into a website. The rate the issuer keeps on a British corporate Visa card used online is 2.00 per cent. On £480 that is **£9.60**.

Look at those two numbers side by side, because the arithmetic is the whole lesson. The ticket costs 114 times as much as the lolly. But the issuer's slice on the ticket is 1,143 times as big. Not 114 times. Eleven hundred. The rate itself went up tenfold, from 0.20 per cent to 2.00 per cent, purely because of what kind of card it was and how it was used, and then that tenfold difference was multiplied by the hundredfold difference in price.

Two things caused that, and neither of them is anything the airline or the coffee shop did.

The first is **what card it was**. A consumer debit card and a corporate credit card are different products with different rules, and the law that holds the consumer card down to 0.20 per cent does not apply to the corporate one at all. Same plastic, same terminal, same two-second shout down the street, ten times the fee.

The second is **how it was used**. Tapped in a shop, with the card physically there, is the cheapest way a transaction can happen, because it is the hardest to fake. Typed into a website is more expensive, because it is easier to fake and somebody has to carry the loss when it is.

And there is a third thing that never shows up in a percentage at all. Almost every merchant's contract has a fixed lump in it — call it five pence a transaction, for the message, the phone line, the reconciliation, the paperwork. Five pence on a £480 ticket is nothing, one hundredth of one per cent. Five pence on a £4.20 lolly is more than the entire interchange fee. The coffee shop is not paying more because coffee is a suspicious business. It is paying more because it sells cheap things, and every cheap thing carries the same fixed lump as an expensive one.

Now the last piece, and it is the piece that decides whether the driver can ever find any of this out.

Mr Doyle can send the driver a bill in one of two shapes. He can write: *forty-one sales, one hundred and seventy-two pounds sixty, our rate one point four per cent plus five pence, total charge seven pounds four*. That is one number, and it is called **blended pricing**, and the driver cannot tell from it whether Mrs Bello took a penny or a pound.

Or Mr Doyle can write: *interchange £1.24, rulebook fees £0.61, our margin £5.19*. Three numbers. That is called **interchange plus plus**, usually written IC++, and it is the only shape of bill in which the driver can see that Mr Doyle's own slice is the biggest of the three.

Which is, of course, exactly why the second shape took a law to make available.

Where the plain version stops being true

The merchant never pays interchange, and the moment of payment is not the moment of the sale. In the picture above, three people each take a slice out of one £4.20 as it travels. That is not what happens. Interchange is never invoiced, never transferred and never paid as a discrete event. It is a *deduction*: when the issuer settles with the scheme for the day's clearing, it remits the transaction value less the applicable interchange, and the acquirer receives less than it presented. Nobody sends anybody a bill. And it is not calculated when the card is tapped — it is calculated at clearing, from the contents of the clearing record, a day or more later. This has a consequence the analogy hides completely: a transaction's interchange rate is not knowable with certainty at the point of sale. It depends on data the merchant supplied, on whether the acquirer presented within the scheme's time limits, and on how the scheme's own rating engine classified the record. A transaction can be re-rated after the fact, and a merchant can be debited weeks later for the difference.

"Scheme fees" is not a fee. It is a catalogue. The single slice taken by Miss Vaughan is, in reality, several hundred separately named line items across two schemes, and they do not share a shape. Some are ad valorem, charged as basis points on transaction value. Some are a fixed amount per authorisation message, charged whether the authorisation was approved or declined. Some are charged per settled transaction, some per chargeback, some per BIN per year, some per licence. Some are charged only when the merchant behaves in a way the scheme wishes to discourage — presenting late, using a terminal that cannot do EMV, exceeding a fraud ratio. Some are charged on refunds. And a large fraction of them are charged not to the acquirer at all but to the *issuer*, which means the merchant never sees them and never pays them, but they still shape the economics of the card in the merchant's

terminal. Any sentence of the form "scheme fees are about X per cent" is a statistical summary of a mess, not a description of a price.

The caps are far narrower than the phrase "regulated interchange" suggests. The 0.2 per cent and 0.3 per cent figures are real, statutory, and — in the UK — apply only where the transaction is a consumer card transaction, on a four-party scheme, with the payer's payment service provider and the payee's payment service provider both located in the United Kingdom. Commercial cards are excluded by the legislation itself. Cash withdrawals are excluded. Three-party schemes are excluded unless they license others. And a transaction where either side of the pair sits outside the UK falls outside the caps entirely, which is not an oversight but the mechanism by which cross-border interchange rose by a factor of five after 2021. "Regulated" describes a subset of British card traffic, not British card traffic.

Blended versus IC++ is not the choice most merchants face, because most merchants have neither. The Payment Systems Regulator found that over 95 per cent of acquirers' merchants are on what it calls *standard* pricing, which is defined by exclusion: it is not IC+, not IC++, and not fixed. Standard pricing is a handful of headline rates attached to broad transaction categories, plus a tail of additional fees triggered by events. It resembles blended pricing in that the merchant cannot see the components, but it is not one blended rate either — it is several, and which one applies to a given sale depends on classifications the merchant was never shown. A merchant asking "am I on blended or IC++?" is usually asking a question with a third answer.

The technical version

The four amounts, and the direction each one travels

There are exactly four money-flows in the fee structure of a four-party card transaction, and the entire subject becomes tractable once their directions are fixed.

Flow	Paid by	Paid to	Mechanism
Interchange fee	Acquirer	Issuer	Deducted at scheme settlement; never invoiced
Scheme fees (acquirer side)	Acquirer	Scheme operator	Periodic invoice, itemised by fee code
Scheme fees (issuer side)	Issuer	Scheme operator	Periodic invoice; invisible to the merchant
Merchant service charge	Merchant	Acquirer	Deducted from funding, or invoiced monthly

The Payment Systems Regulator's formulation, from its card-acquiring market review, is the one to memorise: the MSC "is the total amount it pays for card-acquiring services to its acquirer", and it comprises interchange fees paid by the acquirer to the issuer, scheme fees paid by the acquirer to the scheme operator, and the remainder, which the PSR calls **acquirer net revenue**.

That remainder is the only component the merchant can negotiate, and it is the only component the acquirer keeps. Everything else is pass-through, whether or not the merchant's contract admits it.

The word "scheme fees" in that formulation is doing more work than it appears to. Mastercard and Visa charge acquirers for two logically distinct things: *scheme services*, meaning membership, licensing, brand, rules and dispute machinery; and *processing services*, meaning authorisation switching, clearing

and settlement. The PSR treated these separately in its market review precisely because they behave differently — it concluded that fees for scheme services paid by UK acquirers approximately doubled per pound transacted between 2014 and 2018, while it found insufficient evidence to conclude that processing fees had risen over the same period. A merchant statement that lumps both into one "scheme fee" line is discarding the distinction that matters most.

How interchange is actually determined

Interchange is not negotiated, not quoted and not chosen. It is *derived*, by the scheme's rating engine, from the contents of the transaction record at clearing. The merchant's only influence is over what data reaches that record.

The inputs are, in rough order of importance:

The **issuer's product and country**, taken from the primary account number carried in DE2 and resolved through the BIN to a specific card product — consumer debit, consumer credit, consumer prepaid, business debit, corporate, purchasing, and their platinum and infinite variants — and to an issuing country. A single-digit difference in the BIN changes the applicable fee by an order of magnitude, and the merchant cannot see it.

The **acquirer's country**, which together with the issuer's country determines whether the transaction is domestic, intra-regional or inter-regional. This is the axis on which the regulatory caps live or die.

The **channel and authentication**, expressed in the clearing record through the point-of-service entry mode, the terminal's capability indicators and the presence of EMV data in DE55 for chip transactions, or of an authentication value for e-commerce transactions authenticated through 3-D Secure. Visa's schedules name these tiers *Secure* and *Non-Secure*; Mastercard's name them *Merchant UCAF*, *Full UCAF*, *Enhanced Electronic* and *Base*. Falling out of the top tier is what practitioners call a downgrade, and Visa's own commercial schedules label the bottom tier *Non-Qualified*.

The **merchant category code**, carried in DE18, four numeric digits, the field ISO 8583 names *Merchant type*. The code set is standardised in ISO 18245, *Retail financial services — Merchant category codes*, whose current edition is ISO 18245:2023, superseding the 2003 first edition. MCCs are not decorative. Visa's published UK schedule conditions several programmes on specific codes: the Me-to-Me Program applies to MCC 6012 (financial institutions), MCC 6211 (security brokers and dealers) and MCC 9399 (government services, not elsewhere classified); the tax-payment variant applies to MCC 9311 (tax payments). In Visa's intra-EEA schedule, MCC 8398 (charitable and social service organisations) attracts its own capped commercial rates. A merchant boarded under the wrong MCC pays the wrong interchange indefinitely and will never be told.

The **amount and currency**, from DE4 and DE49 respectively, which matter because a great many rates are percentages subject to a fixed cap, or fixed amounts subject to a percentage floor, and because Visa operates explicit large-ticket programmes with value thresholds.

Finally, **timeliness and data completeness**. Interchange schedules for commercial and purchasing cards pay explicit incentives for enhanced data. Visa's intra-EEA schedule states that "Visa Purchasing acquirers may receive an additional interchange incentive of €0.50 if they submit the transaction with additional data: 'Visa Global Invoicing System and Line Item Detail' ", and that fleet acquirers may receive an additional 0.30 per cent for submitting Level 2 and Level 3 data at fuel-related MCCs. In the UK schedule the equivalents appear as deductions from the headline rate: Visa Purchasing card-not-present is 2.00 per cent, or 2.00 per cent less £0.32 with Line Item Detail, or 2.00 per cent less £0.16 with Summary Tax.

Note what none of these inputs is. None of them is the merchant's size, the merchant's bargaining power, or the merchant's contract. Interchange is identical for the corner shop and the supermarket on identical transactions. Everything that differs between them differs in the other two components.

The published rates, as of August 2026

The schemes publish their interchange schedules. The following are drawn from those published documents; each carries the effective date of the schedule it came from, because these documents are revised on a rolling basis and a figure without a date is a liability.

Mastercard, UK intra-country, consumer, schedule valid from 23 April 2022:

Fee tier	Consumer debit	Consumer credit
Contactless	0.20% (max GBP 1.00)	0.30%
Chip and PIN	0.20% (max GBP 1.00)	0.30%
Merchant UCAF	0.20% (max GBP 1.00)	0.30%
Full UCAF	0.20% (max GBP 1.00)	0.30%
Enhanced Electronic	0.20% (max GBP 1.00)	0.30%
Base	0.20% (max GBP 1.00)	0.30%

The flatness of that table is the regulation showing through. Every consumer tier sits exactly at the statutory ceiling, and the tier structure — which in an unregulated market would spread the rates by a factor of two or three — does nothing at all. The tiers still exist because the same schedule shape is used everywhere; in the UK consumer segment they have been squeezed flat.

Mastercard, UK intra-country, commercial, schedule valid from 22 January 2021, where the same tier structure does its normal work: commercial debit at 0.70 per cent contactless and chip-and-PIN, rising to 1.10 per cent for Enhanced Electronic and Base, with a cap of GBP 1.50 on certain programmes; commercial credit at 1.50 per cent contactless rising to 1.90 per cent Base.

Visa, UK domestic, commercial, schedule effective February 2026:

Product	Card present, EMV incl. contactless	Card not present	Non-qualified
Visa Business Debit	0.75%	1.20%	1.70%
Visa Platinum Business Debit	1.50%	1.80%	2.05%
Visa Infinite Business Debit	1.60%	1.90%	2.05%
Visa Business Credit / Deferred Debit	1.40%	1.70%	2.05%
Visa Platinum Business Credit	1.60%	1.90%	2.05%
Visa Infinite Business Credit	1.70%	2.00%	2.05%
Visa Corporate	1.75%	2.00%	2.05%
Visa Purchasing	1.75%	2.00%	2.05%

Two further rows in that schedule are worth knowing because they are counter-intuitive. *Visa Business Credit — Small Market Expense* is a flat 0.30 per cent, and *Visa Corporate — Large Market Enterprise* is

likewise 0.30 per cent. There exist commercial card programmes that interchange at the consumer credit rate. The generalisation "commercial cards are expensive" is a good default and a bad certainty.

Visa, intra-EEA consumer, schedule effective April 2025, applicable to EEA-issued cards at EEA merchants in a different country: Visa Consumer Debit, Visa Consumer Prepaid, V PAY Debit and V PAY Prepaid at 0.20 per cent across contactless, secure and non-secure; Visa Consumer Credit and Visa Consumer Deferred Debit at 0.30 per cent across the same tiers.

Visa, inter-EEA consumer, effective 19 October 2019, applicable to consumer cards issued outside the EEA transacting at EEA merchants:

Product	Card present	Card not present
Visa Consumer Debit and Prepaid	0.20%	1.15%
Visa Consumer Credit and Deferred Debit	0.30%	1.50%

That last table is the one to hold onto, because those four numbers — 0.20, 1.15, 0.30, 1.50 — recur throughout the regulatory history of the last seven years, and the same four numbers will reappear below in an entirely different context.

The caps: what the law says, and exactly how far it reaches

The instrument is Regulation (EU) 2015/751 of the European Parliament and of the Council of 29 April 2015 on interchange fees for card-based payment transactions, universally called the Interchange Fee Regulation or IFR. Its operative caps took effect on 9 December 2015.

Article 3 provides that "Payment service providers shall not offer or request a per transaction interchange fee of more than 0,2 % of the value of the transaction for any debit card transaction." Article 4 provides the same for credit cards at 0,3 %. Article 3 also permitted Member States to allow a flat per-transaction alternative of no more than EUR 0.05, alone or combined with the percentage, and permitted — until 9 December 2020 only — a domestic weighted-average approach for debit. That transitional route has now expired everywhere; the per-transaction cap is the live rule.

The methodology behind the two numbers is stated in the recitals rather than the articles. Recital 20 records that the caps are based on the Merchant Indifference Test, sometimes called the tourist test: the fee level at which a merchant would be indifferent between being paid by card and being paid in cash, having compared the costs of each. Whatever one thinks of the answer, it is worth knowing that the answer was computed rather than bargained.

The scope limits are in Article 1. Chapter II — which is where the caps live — does not apply to transactions with commercial cards, to cash withdrawals at ATMs or at the counter of a payment service provider, or to transactions with payment cards issued by three-party payment card schemes. Article 1(5) then closes the obvious loophole: where a three-party scheme licenses other payment service providers to issue or acquire, or issues with co-branding partners or through agents, it is considered to be a four-party scheme and the caps bite.

In the United Kingdom the instrument in force is the retained version of that Regulation as amended by the Interchange Fee (Amendment) (EU Exit) Regulations 2019 (SI 2019/284). The territorial scope was rewritten on withdrawal: the retained Regulation applies to card-based payment transactions in the United Kingdom where both the payer's and the payee's payment service providers are located in the United Kingdom. The Payment Systems Regulator is the lead competent authority, designated by HM Treasury, with enforcement powers under the Payment Card Interchange Fee Regulations 2015.

The PSR states the position plainly: consumer cross-border card payments between the UK and the EU, or any other third country, where either the acquirer or the issuer is based outside the UK's jurisdiction, are no longer subject to the interchange fee caps.

Three further articles of the IFR shape the merchant's bill and are routinely forgotten.

Article 5 provides that any agreed remuneration, including net compensation, with an equivalent object or effect to an interchange fee shall be treated as part of the interchange fee. This is the anti-circumvention rule, and it is why a scheme cannot simply route a rebate to issuers by another name.

Article 9 is the unblending rule, and it is the legal foundation of interchange-plus-plus pricing. An acquirer must charge its merchant "merchant service charges individually specified for different categories and different brands", unless the merchant requests blended charges in writing, and the agreement must contain individually specified information on the amount of the merchant service charges, interchange fees and scheme fees for each category and brand. IC++ is not a product the industry invented out of generosity. It is a statutory entitlement that most merchants have contracted out of without noticing.

Article 12 requires that, after execution, the acquirer provides the merchant with a reference enabling the transaction to be identified, the amount, and the charges, "indicating separately the merchant service charge and the amount of the interchange fee". Every merchant in the UK is entitled to transaction-level interchange data. Most have never asked for it.

Article 10 abolishes the honour-all-cards rule across categories — a merchant that accepts a scheme's consumer debit cards is not thereby obliged to accept its commercial cards — and Article 11 prohibits scheme rules that stop merchants steering customers towards a cheaper instrument or telling them what the fees are. Between them, these are the merchant's only real weapons against expensive card types, and both are underused.

All statements in this subsection describe the position as at August 2026. The UK's regulatory architecture is itself in motion: HM Treasury's consultation response of 21 April 2026 confirmed the government's intention to consolidate the Payment Systems Regulator entirely within the Financial Conduct Authority, transferring the PSR's functions including those under the Payment Card Interchange Fee Regulations 2015, subject to primary legislation.

The cross-border hole, which is where the money went

If the caps apply only when issuer and acquirer are both in the UK, then a very large volume of British card traffic sits outside them, and after 2021 the schemes repriced it.

For **UK–EEA** consumer transactions, the PSR's market review of UK-EEA consumer cross-border interchange fees, final report MR22/2.7 published 13 December 2024, records the change precisely. Fees on card-not-present transactions moved from 0.2 per cent to 1.15 per cent for debit and from 0.3 per cent to 1.5 per cent for credit — a fivefold increase on both. Visa announced in March 2021 and implemented in October 2021; Mastercard announced in the third quarter of 2021 and implemented in April 2022. Card-present rates were left at 0.2 and 0.3. The increases applied to what the PSR calls outbound interchange fees, being those on EEA-issued cards used at UK merchants, which is a cost falling on UK merchants and their acquirers. The PSR estimated the additional cost to UK service users at approximately £150 million to £200 million per year.

The PSR found that Mastercard and Visa were not subject to effective competitive constraints and had raised these fees to an unduly high level, and consulted on a two-stage remedy: an interim cap restoring 0.2 per cent for debit and 0.3 per cent for credit on UK-EEA consumer card-not-present transactions,

followed by a lasting cap set by a fuller methodology. As of the review page's October 2025 update, the PSR had decided not to proceed with the interim cap, citing litigation concerning its powers to impose price caps, and was instead consulting on the methodology for a longer-term cap. As of August 2026, therefore, the 1.15 and 1.5 rates remain in force and no UK statutory cap constrains them.

For **inter-regional** transactions — a card issued outside the EEA presented at an EEA merchant — the constraint is not a statute but a competition settlement. On 29 April 2019 the European Commission accepted commitments from Mastercard and Visa, making them legally binding, that cut inter-regional interchange by around 40 per cent on average to the four rates in the table above: 0.2 and 1.15 for debit card-present and card-not-present, 0.3 and 1.5 for credit. Those commitments ran to November 2024, at which point both schemes voluntarily extended the same levels to November 2029, an extension of which the Commission took note. This is a materially weaker form of protection than a cap in a regulation, and it has a stated end date.

The consequence for a merchant is worth stating in one sentence. Two identical £100 online sales, one to a customer holding a British consumer credit card and one to a customer holding a French consumer credit card, generate interchange of 30 pence and £1.50 respectively — five times the fee, for a difference the merchant cannot detect, cannot price for, and cannot decline without breaching its own acquiring agreement.

Scheme fees, named

Scheme fees are opaque by construction, but they are not unknowable, because the PSR's market review of card scheme and processing fees put a number of them on the public record.

On the Visa side, the review documents flat acquiring fees introduced in 2017 — Acquirer Service Fees for card-present and card-not-present — set at 1 basis point for debit transactions and 1.4 basis points for credit. It records the International Acquiring Fee rising from 30 to 45 basis points, and the International Service Assessment Fee rising from 10 to 55 basis points across 2018 to 2020. It records that when UK-EEA traffic became cross-border in 2021, Visa set the International Acquiring Fee on it at 10 basis points for card-present and 25 basis points for card-not-present — a scheme fee increase layered on top of the interchange increase described above, on the same transactions.

On the Mastercard side, the review documents a Switch Pricing Strategy revision of mandatory fees in 2017; a specific card-not-present fee charged to both acquirer and issuer from 2018; the extension of credits and returns scheme fees to credit transactions in 2018, which is the answer to the perennial question of whether refunds are free; a revision of MOTO fee drivers in 2020; a revision of the Acquiring Volume Fee in 2021 that eliminated the minimum volume fee; and mandatory Mastercard Cyber Secure fees in 2021. It also documents behavioural pricing on both sides — Mastercard's non-EMV non-contactless fee from 2019, charged on terminals not meeting the standard, and Visa's Negative Response Fee, removed in 2021 and replaced on the issuing side by a Minimum Approval Rate Integrity Fee. And it documents a structural trick that recurs: services converted from opt-in to opt-out, so that the fee begins to apply unless the participant acts. Mastercard's AAV validation went that way in 2019; Visa's analytics platform likewise.

The aggregate findings are the ones to quote. The PSR's final report MR22/1.10, published 6 March 2025, concluded that Mastercard and Visa had increased their core scheme and processing fees to acquirers by at least 25 per cent since 2017, amounting to at least £170 million extra per year for UK businesses, and that the two schemes were not subject to effective competitive constraints. In its consultation paper CP25/3 of December 2025 the PSR proposed two remedies by direction on Mastercard and Visa: an information, transparency and complexity remedy requiring the schemes to give acquirers clear and accurate information about existing, new and modified fees, with twelve months

to comply after final directions; and a pricing governance remedy requiring pricing decisions to be documented with evidence of how the interests of service users were considered, with four months to comply. Responses were due by 13 February 2026, with a third remedy on regulatory financial reporting deferred to a consultation by 31 March 2026.

Note what is being remedied. Not the level of the fees. The *legibility* of the fees.

The acquirer's share, and everything else on the invoice

Acquirer net revenue is defined by subtraction: MSC less interchange less scheme fees. It is the price of the acquirer's own service, and it covers the settlement risk the acquirer carries between funding the merchant and being made whole, the chargeback exposure, the underwriting, the reconciliation, the support, and the capital.

For the five largest UK acquirers — Barclaycard, Elavon, Global Payments, Lloyds Bank Cardnet and Worldpay — the PSR found that acquirer net revenues for card-acquiring services accounted for 62 per cent of total revenues, with card acceptance devices and payment gateways contributing 15 per cent and value-added services the remaining 23 per cent. Slightly more than a third of what a merchant pays its acquirer, in aggregate, is not for acquiring at all.

That surrounding third arrives as a tail of additional fees, and it is where the effective rate of a small merchant is actually decided. The PSR's description of standard pricing is worth reproducing in substance: several headline rates applied to different types of purchase transaction, each of which can be a pence-per-transaction fee, an ad valorem fee or a combination; plus one or more additional fees triggered either by specific events such as chargebacks, refunds and PCI DSS non-compliance, or by specific transaction types such as e-commerce. Add to this the minimum monthly service charge, which some acquirers apply when a merchant's monthly spend falls below a threshold; the monthly fee for PCI DSS compliance services; and terminal hire, which the PSR put at typically £10 to £40 per month per device, against card readers sold outright by the largest payment facilitators for between £15 and £45.

A merchant taking £2,000 a month through a single terminal, paying £25 for the terminal, £10 for PCI compliance and a £20 minimum service charge, is paying £55 in non-transactional fees. On £2,000 of turnover that is 2.75 per cent before a single transaction is rated. This is the arithmetic that explains why the very smallest merchants overwhelmingly use payment facilitators with a single headline rate and no monthly fees, even though that headline rate is higher: the PSR found the largest payment facilitators serve nearly 80 per cent of merchants with annual card turnover up to £15,000, and fewer than 5 per cent of small and medium merchants with turnover above £60,000. The crossover is real and it is arithmetic, not marketing.

The four pricing models, stated exactly

The PSR's taxonomy is the authoritative one and the four definitions are worth having verbatim in substance.

Standard pricing is pricing where, for any given transaction, the acquirer does not automatically pass through at cost the interchange fee applicable to that transaction, and which does not satisfy the criteria for IC+, IC++ or fixed pricing.

Interchange fee plus (IC+) is pricing where, for any given transaction, the acquirer automatically passes through at cost the interchange fee applicable to that transaction. Scheme fees remain inside the acquirer's margin.

Interchange fee plus plus (IC++) is pricing where, for any given transaction, the acquirer automatically passes through at cost both the interchange fee and the scheme fees applicable to that transaction.

Fixed pricing is where the merchant pays a fixed periodic fee for card-acquiring services whose amount does not depend on the volume or value of transactions accepted or their characteristics, within specified limits; the acquirer may separately pass through at cost any scheme fees not directly attributable to transactions.

Over 95 per cent of acquirers' merchants are on standard pricing. IC++ is concentrated at the top of the market, and the PSR estimated that for merchants with annual card turnover above £50 million on IC++ pricing, the benefit of the IFR savings was around £600 million in 2018 — because on IC++ a cut in interchange reaches the merchant automatically. For small and medium merchants the same review found, on average, little or no pass-through of the IFR savings, while acquirers may have passed on nearly all of the scheme fee increases. The pricing model is not a presentational choice. It determines who keeps the benefit of regulation and who absorbs the cost of scheme repricing.

Why the coffee shop and the airline pay different effective rates

Assemble the pieces and the answer has five independent components, none of which is "the airline negotiated better", although the airline did.

Ticket size. Fixed per-transaction elements — authorisation fees, per-item scheme fees, the pence component of a headline rate — are constant in pence and therefore hyperbolic in percentage terms. A 5p fixed element is 1.19 per cent of a £4.20 coffee and 0.01 per cent of a £480 fare. Nothing else in the stack produces a hundredfold difference this cheaply.

Card mix. The coffee shop's traffic is overwhelmingly UK consumer debit, capped at 0.20 per cent. The airline's traffic is heavy in commercial and corporate cards — the very products the IFR excludes — where Visa's published UK card-not-present rate for Visa Corporate is 2.00 per cent. A merchant whose customers pay with company cards is structurally, permanently more expensive to serve, and can do nothing about it except invoke Article 10 and refuse the category, which no airline will do.

Channel. The coffee shop is card-present, contactless, EMV, in the top interchange tier and the lowest fraud-liability position. The airline is card-not-present, which raises interchange on cross-border and commercial traffic, attracts card-not-present scheme fees on both sides, carries higher chargeback exposure, and — for cross-border consumer cards — is precisely the channel where UK-EEA interchange went to 1.15 and 1.5 per cent.

Region. The coffee shop's customers are almost all UK-issued. The airline's are, by the nature of the business, spread across the EEA and the rest of the world, which means a material fraction of its interchange is not capped at all, plus international acquiring and service assessment fees on top.

Merchant category code. The airline's MCC signals a delayed-delivery business: the customer pays in March and flies in August, and if the airline fails in June the acquirer is exposed to every unflown ticket. That risk is priced in acquirer net revenue and in collateral, not in interchange, and it is why the PSR observed that Elavon specialises in serving airlines, "which carry a higher credit risk than many other merchants". The airline's low headline rate frequently coexists with a rolling reserve that costs it more in working capital than the whole MSC.

Size, last and least interesting. The PSR's segmentation shows why bargaining power concentrates where it does: merchants with annual card turnover above £50 million are 0.1 per cent of merchants and 76 per cent of transaction value, while merchants below £380,000 are 93.7 per cent of merchants

and 6.5 per cent of value. The PSR found the market working adequately for the first group and not working well for everyone below £50 million.

One transaction, priced twice

The following puts the whole chapter into two columns. The interchange figures are published scheme rates as cited above. The scheme fee and acquirer margin figures are illustrative of a plausible contract and are marked as such — they are not published rates, and no merchant should treat them as a benchmark.

	Coffee shop	Airline
Transaction	£4.20	£480.00
Card	UK consumer debit	UK Visa Corporate
Channel	Contactless, card present	E-commerce, card not present
Interchange rate (published)	0.20%	2.00%
Interchange amount	£0.0084	£9.60
Scheme fees (illustrative)	£0.009	£0.53
Acquirer net revenue (illustrative)	£0.092	£0.54
Total MSC	£0.109	£10.67
Effective rate	2.60%	2.22%

And now the twist that makes the table worth printing. On these assumptions the airline's effective rate is *lower* than the coffee shop's, but barely — and the reason is not that the airline is being overcharged. It is that the airline's customer used a corporate card. Rerun the same £480 sale on a UK consumer credit card at 0.30 per cent and the interchange falls from £9.60 to £1.44, the MSC falls to about £2.51, and the effective rate falls to 0.52 per cent. Same merchant, same channel, same acquirer, same day, five times the difference, decided entirely by which piece of plastic came out of the wallet.

The coffee shop cannot reach 0.52 per cent by any means available to it, because at £4.20 a transaction the fixed components dominate everything else. The airline reaches it or does not reach it depending on its customer mix, which it does not control either.

Which is the honest ending to this chapter. Almost none of what a merchant pays is set by the merchant, or by the party that invoices the merchant, or in the country the merchant trades in. It is set in a rate schedule published by a company the merchant has no contract with, applied to a card issued by a bank the merchant has never dealt with, through a rating engine the merchant cannot see, a day after the sale. The only two levers a merchant genuinely holds are the acquirer's own margin, which is typically the smallest of the three components on a large ticket and the largest on a small one, and the shape of the bill — because a merchant who cannot see the three components separately cannot tell which lever to pull.

That is what Article 9 was for. Ask for it in writing.

31.98 Common wrong ideas

Wrong: The merchant pays interchange to Visa or Mastercard. **Right:** The acquirer pays it to the card-issuing bank; the scheme is not the recipient and the merchant is not the payer, and the merchant has no relationship with the party that receives it.

Wrong: Interchange is a fee that somebody invoices. **Right:** It is a deduction taken when the issuer settles with the scheme for the day's clearing, so no bill for it is sent to anybody and no discrete transfer of it ever happens.

Wrong: The interchange rate is fixed at the moment the card is tapped. **Right:** It is derived at clearing from the contents of the clearing record, so a transaction can be re-rated afterwards and a merchant debited weeks later for the difference.

Wrong: Scheme fees are a percentage you can quote. **Right:** They are several hundred separately named line items across two schemes, some ad valorem, some per authorisation message whether approved or declined, some per chargeback, per BIN or per licence, some charged only on behaviour the scheme wishes to discourage, and many charged to the issuer where the merchant never sees them.

Wrong: UK interchange is regulated, so the caps apply to UK card payments. **Right:** The caps apply only to consumer card transactions on four-party schemes where the payer's and the payee's payment service providers are both located in the United Kingdom, which leaves commercial cards, cash withdrawals and every cross-border pair outside them.

Wrong: Commercial cards are always expensive. **Right:** Visa Business Credit Small Market Expense and Visa Corporate Large Market Enterprise both interchange at 0.30 per cent, the consumer credit rate, so the generalisation is a good default and a bad certainty.

Wrong: A merchant chooses between blended pricing and IC++. **Right:** Over 95 per cent of merchants are on standard pricing, which is neither: several headline rates attached to broad categories, plus a tail of event-triggered fees, with the classifications that decide which rate applies never shown to the merchant.

Wrong: Large merchants get better interchange than small ones. **Right:** Interchange is identical for the corner shop and the supermarket on identical transactions, and everything that differs between them differs in the acquirer's margin and the scheme fees.

Wrong: IC++ pricing is a concession an acquirer may or may not grant. **Right:** Article 9 makes individually specified charges the default entitlement, and blending requires the merchant to ask for it in writing, which is a statutory right most merchants have contracted out of without noticing.

Wrong: A low headline rate means a cheap merchant account. **Right:** Terminal hire, a PCI compliance fee and a minimum monthly service charge can amount to 2.75 per cent of a small merchant's turnover before a single transaction has been rated, which is the arithmetic that sends the smallest merchants to payment facilitators.

31.99 Chapter summary in 20 lines

1. The gap between what a customer pays and what a merchant receives is called the merchant service charge, and it is three fees wearing one coat.
2. Interchange runs from the acquirer to the issuing bank, which is the opposite direction from the one almost everybody assumes.
3. Scheme fees go from both the acquirer and the issuer to Visa or Mastercard, for the rulebook, the brand, the dispute machinery and the switching.
4. Acquirer net revenue is what is left after those two, and it is the only component the merchant can negotiate.
5. Interchange is never invoiced: the issuer remits the transaction value less the applicable fee, and the acquirer simply receives less than it presented.
6. It is also derived rather than quoted, computed by the scheme's rating engine at clearing, which means it is not certain at the point of sale and can be re-rated later.

7. The inputs are the issuer's product and country, the acquirer's country, the channel and authentication tier, the merchant category code, the amount and currency, and the completeness of the data submitted.
8. None of those inputs is the merchant's size, its bargaining power or its contract, so interchange is the same for the corner shop and the supermarket.
9. The statutory caps of 0.2 and 0.3 per cent come from the Interchange Fee Regulation and were computed by the Merchant Indifference Test rather than bargained.
10. Those caps reach only consumer cards on four-party schemes with both payment service providers in the United Kingdom, excluding commercial cards, cash withdrawals and all cross-border traffic.
11. That exclusion is where the money went: UK-EEA consumer card-not-present interchange moved from 0.2 to 1.15 per cent for debit and 0.3 to 1.5 per cent for credit, at an estimated cost to UK service users of £150 million to £200 million a year.
12. Inter-regional rates sit at the same four numbers, but by a competition settlement with a stated end date rather than by statute, which is materially weaker protection.
13. Scheme fees are not one price but a catalogue of several hundred items of different shapes, some charged on declines, some on refunds, and many charged to issuers where the merchant never sees them but still feels the economics.
14. The regulator found scheme and processing fees to acquirers up by at least 25 per cent since 2017, worth at least £170 million a year, and proposed remedies aimed at the legibility of those fees rather than their level.
15. Over 95 per cent of merchants are on standard pricing, in which the components are invisible and the applicable headline rate depends on classifications the merchant was never shown.
16. Article 9 makes unblended charges a statutory entitlement and Article 12 entitles every merchant to transaction-level interchange data, and most merchants have never asked for either.
17. Slightly more than a third of what merchants pay their acquirers in aggregate is not for acquiring at all, but for devices, gateways and value-added services.
18. Fixed per-transaction elements are constant in pence and therefore hyperbolic in percentage terms, which is why the coffee shop's effective rate can exceed the airline's.
19. Rerun the same £480 sale on a consumer credit card rather than a corporate card and the effective rate falls from about 2.22 per cent to about 0.52 per cent, decided entirely by which piece of plastic came out of the wallet.
20. Almost nothing a merchant pays is set by the merchant, by the party that invoices the merchant, or in the country the merchant trades in, so the only genuine levers are the acquirer's own margin and the shape of the bill.

Sources: Regulation (EU) 2015/751 (Articles 1, 3, 4, 5, 9, 10, 11, 12 and Recital 20) as retained in UK law and amended by the Interchange Fee (Amendment) (EU Exit) Regulations 2019; Payment Systems Regulator MR18/1.8, MR22/1.6, MR22/1.10, CP25/3 and MR22/2.7 and its IFR guidance pages; European Commission press release IP/19/2311; published interchange schedules of Visa (UK, intra-EEA and inter-EEA) and Mastercard (UK intra-country); ISO 8583 and ISO 18245:2023; HM Treasury, "A Streamlined Approach to Payment Systems Regulation" consultation response; UK Finance card spending statistics.

10

CHAPTER

ISO 20022

32.0 What this chapter gives you

1. You will be able to explain why ISO 20022 is not a format but a method for producing formats, a dictionary of business concepts and a committee structure for deciding what goes into it.
2. You will be able to say what actually retired on 22 November 2025, and why the authorisation message leaving a shop terminal was untouched by it.
3. You will be able to read a message identifier such as `pacs.008.001.08`, name each of its four parts, and say why a version increment is a release rather than a configuration change.
4. You will be able to distinguish a `pain` message from a `pacs` message and say which party sends which.
5. You will be able to explain why "richer data" describes a capacity rather than a delivery, given that the unstructured remittance field is the same one hundred and forty characters in both the old message and the new one.
6. You will be able to say what a structured postal address changes for sanctions screening, and why the address mandate slipped repeatedly for reasons that were about stored data rather than software.
7. You will be able to explain why translating a rich message into a legacy format is lossy by construction, and why a round-trip diff is a more useful document than any readiness report.
8. You will be able to say why validating a message against its schema does not validate its codes, and how often the external code sets have to be refreshed.
9. You will be able to explain why a domestic real-time gross settlement system can cut over in a weekend while a correspondent network of eleven thousand institutions needed two and a half years of coexistence.
10. You will be able to state the single design rule the chapter turns on: keep the ISO 20022 message as the system of record, and translate only at the last mile.

The name is a trap. Everything else in this volume has been a format: ISO 8583 is a format, EMV tag-length-value is a format, a Bacs Standard 18 record is a format. ISO 20022 is not a format. It is a method for producing formats, plus a dictionary of business concepts that those formats draw on, plus a committee structure for deciding what goes into the dictionary. When somebody says "we've gone ISO 20022", they have not adopted a message. They have adopted a family of several hundred messages, of which they probably use four.

The second trap is the word "replaces". ISO 20022 did not replace ISO 8583 in the way that HTTP/2 replaced HTTP/1.1. As of August 2026, when you tap a card in a shop, the authorisation that leaves the terminal is still an ISO 8583 `0100` and the reply is still an `0110`, and there is no serious industry programme to change that. What ISO 20022 has actually replaced, and replaced almost completely, is

the *other* legacy format: the SWIFT MT messages that carried cross-border and high-value payments between banks. That migration finished its main phase on 22 November 2025. This chapter is about what changed, what it bought, what it cost, and why a change that everybody agreed on in 2004 was still being argued about in 2027.

The plain version

In the ISO 8583 chapter you learned the numbered form: everybody memorises the code book, box 4 is always the amount, and the message is short because nothing in it says what anything is. That works beautifully until the day you need to put something in the message that the code book never anticipated.

So think about a freight depot instead.

Harkness Joinery in Leeds sends pallets of timber all over Europe. For forty years every pallet went out with a docket: a small card with numbered boxes on it, printed in the depot's own house style. Box 3 is the weight. Box 8 is the destination depot code. Box 14 is *notes*, and box 14 is one hundred and forty characters long, no more, because that is how wide the card is.

The docket works. It is small, it is cheap, it goes through the sorting machine at speed, and every depot from Leeds to Lisbon has the code book pinned to the wall. The trouble is box 14.

Because what actually needs to travel with the pallet is not a note. It is a list. This pallet covers three orders — order 4471 for £7,200, order 4488 for £6,150 and order 4502 for £5,100 — and the customer's warehouse clerk needs to know which planks belong to which order, or she cannot sign anything off. One hundred and forty characters is not enough for three orders with their dates and amounts, so the Leeds despatcher does what despatchers everywhere do. He writes:

```
ORD 4471 4488 4502 PART DEL SEE EMAIL
```

and the pallet leaves. At the other end a human being reads that, opens a spreadsheet, and spends twenty minutes working out what arrived. Multiply by four hundred pallets a day.

There is a second problem, and it is quieter. Box 9 on the docket is the consignee: four lines of text, thirty-five characters each. The Leeds despatcher types:

```
VAN DOORN HOUT BV
KANAALDIJK 14
5613 EA EINDHOVEN
NETHERLANDS
```

That is perfectly clear to you. It is completely opaque to a machine. Nothing on the docket says which of those four lines is the town. A computer at a border checkpoint that has been told to stop every consignment going to one particular address in one particular city has to guess, by pattern-matching lumps of text against a list, and it will guess wrong in both directions: it will stop pallets it should let through, and let through pallets it should stop.

Now the new sheet.

The new sheet is A4, not a card. Every line on it says what it is. Instead of box 9 there is a block that reads:

```
STREET NAME: Kanaaldijk
BUILDING NUMBER: 14
POST CODE: 5613 EA
```

```
TOWN NAME: Eindhoven
COUNTRY: NL
```

and instead of box 14 there is a whole labelled section headed *what this pallet settles*, which repeats as many times as it needs to:

```
ORDER NUMBER: 4471   DATED: 2026-06-02   AMOUNT: 7,200.00
ORDER NUMBER: 4488   DATED: 2026-06-09   AMOUNT: 6,150.00
ORDER NUMBER: 4502   DATED: 2026-06-15   AMOUNT: 5,100.00
```

Three things follow from that, and they are the whole point of the standard.

The sheet is fatter. It is several times the size of the docket, because every value now drags a label along behind it. In 1985 that would have been disqualifying. In 2026, when the depot's network cost is not measured in characters, it is a rounding error.

The sheet is self-describing. A depot that has lost the code book can still read it. More usefully, a piece of software written by someone who has never met the Leeds despatcher can read it, because "TOWN NAME" means town name everywhere the sheet is used, in every kind of sheet the depot prints — deliveries, returns, invoices, insurance claims. That consistency is not a happy accident; it is the actual product. Somebody sat down and wrote a dictionary that says what a town name is and what a consignee is, and then every form gets built out of the dictionary rather than being invented from scratch.

The clerk at the far end stops doing twenty minutes of work. This is what "richer data" means, stripped of the brochure language. It does not mean more characters. It means the same facts arrive already separated into their component parts, so that a machine can act on each part without guessing.

Now translate the whole thing back into money, with real numbers.

Harkness Joinery pays Van Doorn Hout BV **€18,450.00**, settling those three invoices. Under the old system that payment travelled as an MT 103, and the MT 103 has a field for remittance information called field 70, and field 70 is four lines of thirty-five characters — **one hundred and forty characters** — and that is all there is. The beneficiary's name and address travel in field 59, four lines of thirty-five characters, undifferentiated. Van Doorn's finance team gets a bank statement showing €18,450.00 from a name they half recognise and a reference that reads `INV 4471 4488 4502`. Somebody matches it by hand.

Under the new system the same payment travels as a message called `pacs.008`. The three invoices go in as three separate structured entries, each with its own number, its own date and its own amount. The address goes in with the town in the box marked town and the country in the box marked country. Van Doorn's ledger software reads the message, matches all three invoices automatically, closes them, and nobody touches it.

That is the promise. Here is why it took twenty-two years to deliver.

The new sheet is worthless unless the depot at the other end can read it. If Leeds prints the A4 sheet and Eindhoven only has a slot for the small card, the sheet has to be cut down to fit — and cutting it down means throwing away exactly the three invoice lines that made it worth printing. So for years every depot has to be able to handle both, and the network only moves at the speed of its worst reader. Nobody can go first, because going first costs money and buys nothing. Everybody has to go at once, and "everybody" is eleven thousand institutions in two hundred countries, each with thirty years of software behind the counter that was written on the assumption that a reference is never longer than thirty-five characters.

The way that gets solved, in the end, is not persuasion. It is a date. On **22 November 2025** the old dockets stopped being accepted for cross-border payment instructions on the SWIFT network, and everybody who had been putting it off found out what their real deadline had been all along.

Where the plain version stops being true

ISO 20022 is not a message format, and calling it "the replacement for ISO 8583" is wrong in two directions at once. It is a modelling standard: a metamodel, a repository of business concepts, and a set of rules for generating machine-readable schemas from that repository. The catalogue currently holds around eight hundred registered message definitions spanning payments, securities, foreign exchange, trade finance, collateral and cards. What was retired on 22 November 2025 was not ISO 8583 but SWIFT MT — the FIN messages of the ISO 15022 era — for cross-border payment instructions and reporting. Card authorisation traffic is untouched. ISO 20022 does contain card message sets (the business areas `caaa`, `cain`, `catp` and `casp` cover acceptor-to-acquirer, acquirer-to-issuer, ATM and sale-to-POI messaging, and the nexo standards build on them), and they have made almost no impression on global acquiring volume. The `0100` you met earlier in this volume will outlive everyone reading this.

"Richer data" describes a capacity, not a delivery. The unstructured remittance field in `pacs.008` is still one hundred and forty characters, exactly the same as MT 103 field 70; the CBPR+ market practice for cross-border payments permits one occurrence of it. The gain is structure, not length, and structure only pays when the sender populates it and the receiver consumes it. The clearest evidence that this is harder than it sounds is the Bank of England: having proposed to mandate structured remittance data in CHAPS from November 2027, it announced in **July 2026** that it would not, and would instead work on international alignment first. A field that exists, is schema-valid and is uniformly empty has bought nobody anything.

XML versus packed binary is the least interesting difference between the two standards, and ISO 20022 is not intrinsically XML. ISO 20022-4:2013 specifies XML Schema generation; ISO 20022-8:2013 specifies ASN.1 generation; the Registration Management Group approved a JSON Schema generation recommendation on 10 June 2025 targeting JSON Schema Draft 2020-12; and ISO 20022-9, published in April 2026, generalises syntax generation rules so that the model is not bound to any one encoding. Card messages under nexo are routinely carried as ASN.1, not XML. The substantive difference is not angle brackets versus bitmaps. It is that ISO 8583 defines a message and leaves the meaning of its contents to bilateral agreement, whereas ISO 20022 defines the meaning centrally and derives the message from it. That is why two ISO 8583 implementations can both be "compliant" and unable to talk to each other, and why the same concept has the same name in an ISO 20022 payment, statement and investigation.

Coexistence and translation preserve delivery, not meaning, and the loss is arithmetic rather than sloppiness. During coexistence a rich `pacs.008` bound for a bank that could only receive MT was translated in flight. MT 103 field 70 holds 140 characters and field 59 holds four lines of thirty-five; a `RemittanceInformation` block with three structured invoice entries and a fifteen-element postal address does not fit, so it is truncated or dropped. The translation is not reversible: converting back the other way produces a schema-valid message containing less than the original. This matters legally as well as operationally, because the originator and beneficiary information that must accompany a wire transfer under FATF Recommendation 16 is precisely the information that translation squeezes. An institution that keeps a translated MT as its record of the payment has quietly destroyed the evidence the standard existed to carry.

The technical version

What the standard actually is

ISO 20022 is maintained under ISO Technical Committee 68, Subcommittee 9. The core standard was published in eight parts in May 2013 and extended in 2026:

Part	Title	Published
ISO 20022-1	Metamodel	May 2013
ISO 20022-2	UML profile	May 2013
ISO 20022-3	Modelling	May 2013
ISO 20022-4	XML Schema generation	May 2013
ISO 20022-5	Reverse engineering	May 2013
ISO 20022-6	Message transport characteristics	May 2013
ISO 20022-7	Registration	May 2013
ISO 20022-8	ASN.1 generation	May 2013
ISO 20022-9	Syntax generation requirements and rules	April 2026

The architecture has three layers and confusing them is the source of most arguments about the standard. At the top is the **business model**: business processes, business roles and business concepts, expressed in UML and independent of any message. In the middle is the **logical message**: a message definition assembled from components in the Data Dictionary. At the bottom is the **physical syntax**: an XML Schema, an ASN.1 module or a JSON Schema, generated mechanically from the logical message by the rules in Parts 4, 8 or 9. The generated XSDs carry a comment recording the Standards Editor build and the ISO 20022 version they were generated from, which tells you they are outputs, not hand-written artefacts.

Governance is a registry, not a standards body in the usual sense. SWIFT acts as **Registration Authority**, maintaining the repository and checking that submissions comply with the technical specifications. The **Registration Management Group** oversees registration and reports to TC 68/SC 9. **Standards Evaluation Groups** — for Payments, Securities, Cards and Related Retail Financial Services, Foreign Exchange, Trade Finance and API Resources — validate new message definitions from a business perspective and approve changes to existing ones. A **Cross-SEG Harmonisation Group** resolves conflicts where the same concept appears in more than one domain, and a **Technical Support Group** handles the syntax mechanics. A new message begins with a business justification accepted by the RMG, is developed by a submitting organisation, is validated by the relevant SEG, and is then published by the RA into the repository.

Message definitions are grouped into four-letter business areas. The payments and cards ones matter here:

Code	Business area
`pain`	Payments initiation
`pacs`	Payments clearing and settlement
`camt`	Cash management
`remt`	Payments remittance advice
`acmt`	Account management
`admi`	Administration
`head`	Business application header
`auth`	Authorities

Code	Business area
reda	Reference data
caaa	Acceptor to acquirer card transactions
cain	Acquirer to issuer card transactions
catp	ATM card transactions
casp	Sale to POI card transactions
catm	POI management
tokm	Payment token management

The distinction between `pain` and `pacs` is the one that catches newcomers. A `pain` message is customer to bank: a corporate instructing its bank to make payments. A `pacs` message is bank to bank: the interbank leg. A corporate never sends a `pacs.008` and a clearing system never receives a `pain.001`.

The message identifier and the namespace

A message definition identifier has four parts:

```
pacs . 008 . 001 . 08
 |      |     |     |
 |      |     |     +-- version (two digits)
 |      |     +-------- variant (three digits; 001 is the base variant)
 |      +-------------- message functionality (three digits)
 +--------------------- business area (four letters)
```

`pacs.008.001.08` is version 8 of the base variant of message 008 in the payments clearing and settlement area — `FIToFICustomerCreditTransferV08`. The XML target namespace embeds the whole identifier:

```
urn:iso:std:iso:20022:tech:xsd:pacs.008.001.08
```

That last point has an operational consequence that is regularly underestimated. A version increment changes the namespace, which changes every schema binding, every XPath expression and every generated class in every application that touches the message. Upgrading from `pacs.008.001.08` to `pacs.008.001.13` is not a configuration change. It is a release. This is why market infrastructures pin versions and hold them for years: the SEPA 2025 scheme implementation guidelines use the 2019 message versions throughout — `pacs.008.001.08`, `pacs.004.001.09`, `pacs.002.001.10`, `pacs.028.001.03`, `camt.056.001.08`, `camt.029.001.09`, `camt.027.001.07`, `camt.087.001.06` — and CBPR+ pinned the same generation for cross-border.

The MT messages that were retired map across roughly as follows. "Roughly" is load-bearing: the mappings are many-to-one in places and none of them is lossless.

Retired MT	ISO 20022 equivalent
MT 103, MT 102	`pacs.008` FI to FI customer credit transfer
MT 200, 201, 202, 203, 205	`pacs.009` financial institution credit transfer
MT 202 COV	`pacs.009` COV variant
MT 900, MT 910	`camt.054` debit/credit notification
MT 940, MT 950	`camt.053` bank to customer statement
MT 941, MT 942	`camt.052` bank to customer account report
MT 192, MT 292	`camt.056` payment cancellation request
MT 196, MT 296	`camt.029` resolution of investigation
MT 101	`pain.001` customer credit transfer initiation

MT 199 and MT 299 were not retired in November 2025; they remain in use for transaction tracking, with their exceptions-and-investigations and payment-initiation use cases scheduled to migrate later.

Anatomy of a pacs.008

A CBPR+ payment on the wire is two documents in one envelope. The first is the **Business Application Header**, `head.001.001.02`, which carries routing and identification: `Fr` and `To` (the sending and receiving BICs), `BizMsgIdr` (the business message identifier, which CBPR+ requires to equal the group header's `MsgId`), `MsgDefIdr` (`pacs.008.001.08`), `BizSvc` (the business service, for example `swift.cbprplus.02`, or `swift.cbprplus.stp.02` for the straight-through-processing variant) and `CreDt`.

The second is the message document itself, rooted at `FIToFICstmrCdtTrf`, with a `GrpHdr` and one or more `CdtTrfTxInf`. Under CBPR+, `NbOfTxs` is fixed at 1 — one payment per message — which is a market practice restriction, not a schema restriction, and is the single most common cause of a schema-valid message being rejected by the network.

The datatypes are worth stating exactly, because they are where the contrast with ISO 8583 becomes concrete. Taking the definitions from the `pacs.008.001.09` XML Schema:

- `PmtId/EndToEndId` is `Max35Text`, mandatory, and must be passed unchanged along the whole chain.
- `PmtId/UETR` is `UUIDv4Identifier`, constrained by the pattern `[a-f0-9]{8}-[a-f0-9]{4}-4[a-f0-9]{3}-[89ab][a-f0-9]{3}-[a-f0-9]{12}` — thirty-six characters, a version-4 UUID, unique for the life of the payment.
- `IntrBkSttlmAmt` is `ActiveCurrencyAndAmount`: an `xs:decimal` restricted to `totalDigits` 18, `fractionDigits` 5 and `minInclusive` 0, carrying a mandatory `Ccy` attribute matching `[A-Z]{3,3}` — the ISO 4217 alphabetic code.
- `ChrgBr` is `ChargeBearerType1Code`, an enumeration of exactly `DEBT`, `CRED`, `SHAR` and `SLEV`.
- `SttlmInf/SttlmMtd` is `SettlementMethod1Code`: `INDA`, `INGA`, `COVE`, `CLRG`, `TDSO`, `TDSA`.
- Agent identification uses `BICFI`, typed `BICFIDec2014Identifier` with the pattern `[A-Z0-9]{4,4}[A-Z]{2,2}[A-Z0-9]{2,2}([A-Z0-9]{3,3}){0,1}` — the ISO 9362 BIC, eight or eleven characters.
- Accounts use `IBAN`, typed `IBAN2007Identifier`, pattern `[A-Z]{2,2}[0-9]{2,2}[a-zA-Z0-9]{1,30}` — ISO 13616.
- Legal entities use `LEI`, typed `LEIIdentifier`, pattern `[A-Z0-9]{18,18}[0-9]{2,2}` — twenty characters, ISO 17442.
- `Ctry` is `CountryCode`, pattern `[A-Z]{2,2}` — ISO 3166-1 alpha-2.
- `CreDtTm` is `ISODateTime`, which is `xs:dateTime`; CBPR+ requires the UTC offset to be present.

The same facts in both formats

This is the table to keep. It is the honest comparison between what the terminal sends and what the interbank leg sends, for the concepts that appear in both.

Concept	ISO 8583	ISO 20022 (`pacs.008`)
Message type	MTI, four digits: version, class, function, origin (`0100`, `0200`, `0800`)	Message definition identifier plus namespace (`pacs.008.001.08`)
Which fields are present	64-bit primary bitmap; bit 1 signals a secondary bitmap for elements 65 to 128	Element presence in the XML instance; cardinality in the schema
Account identifier	DE2 PAN	`DbtrAcct/Id/IBAN` or `Othr/Id`

Concept	ISO 8583	ISO 20022 (`pacs.008`)
Transaction nature	DE3 processing code	`PmtTpInf`, plus `Purp/Cd` from an external code set
Amount	DE4, fixed 12 digits, implied decimal from the currency's minor units	`IntrBkSttlmAmt`, decimal, 18 total digits, 5 fractional
Currency	DE49, ISO 4217 numeric, separate field	`Ccy` attribute on the amount, ISO 4217 alphabetic
Date and time	DE7 transmission date and time, `MMDDhhmmss`, no year, no zone; DE12 local time, DE13 local date	`CreDtTm`, `xs:dateTime` with offset; `IntrBkSttlmDt`, `xs:date`
Trace number	DE11 STAN, six digits, wraps within the day	`PmtId/UETR`, UUIDv4, unique end to end
Reference	DE37 retrieval reference number, twelve characters	`PmtId/EndToEndId`, `Max35Text`; `PmtId/TxId`; `PmtId/InstrId`
Approval identifier	DE38 authorisation identification response, six characters	No equivalent; the model is not authorise-then-clear
Outcome	DE39 response code, two characters	`pacs.002` `TxSts`, four characters, plus `StsRsnInf/Rsn/Cd`
Terminal	DE41 terminal ID, eight characters	No equivalent
Chip data	DE55, EMV tag-length-value	No equivalent

Two rows in that table are worth dwelling on.

The **amount** row explains a class of migration bug. DE4 is a fixed-width integer whose decimal point is implied by the currency in DE49; `18450.00` in euros is the twelve characters `000001845000`. `IntrBkSttlmAmt` is a decimal with the currency as an attribute: `<IntrBkSttlmAmt Ccy="EUR">18450.00</IntrBkSttlmAmt>`. Anything that parses the second into a binary floating-point type has introduced a defect that will surface as a one-penny break in a reconciliation six months later. Five fractional digits also means the schema will happily accept `18450.00001`, which is a valid decimal and not a valid euro amount; the constraint that the value must respect the currency's minor units lives in the usage guideline, not in the XSD.

The **outcome** row explains why anyone bothered. In ISO 8583, everything the issuer wants to say about a decline is compressed into DE39, two characters, of which the most commonly used means "do not honour" and is deliberately uninformative. In ISO 20022 the outcome is split: a `pacs.002` carries `TxSts` from a four-character external code set — `RCVD`, `ACTC`, `ACCP`, `ACSP`, `ACWC`, `ACSC`, `PDNG`, `RJCT` — and where the status is `RJCT` a separate `StsRsnInf/Rsn/Cd` is mandatory, itself from a further external code set, with `AddtlInf` available for free text. Status and reason are different facts and get different fields. Whether the reason is any more candid than "do not honour" is a policy question, not a format question, and in the sanctions cases it usually is not.

The address problem, exactly

Structured addresses are the least glamorous part of this migration and by a distance the most expensive. Here is the actual structure, from the `pacs.008.001.09` schema (the complex type is `PostalAddress24`):

Element	Tag	Type
Address type	`AdrTp`	choice
Department	`Dept`	`Max70Text`
Sub department	`SubDept`	`Max70Text`
Street name	`StrtNm`	`Max70Text`
Building number	`BldgNb`	`Max16Text`
Building name	`BldgNm`	`Max35Text`
Floor	`Flr`	`Max70Text`
Post box	`PstBx`	`Max16Text`
Room	`Room`	`Max70Text`
Post code	`PstCd`	`Max16Text`
Town name	`TwnNm`	`Max35Text`
Town location name	`TwnLctnNm`	`Max35Text`
District name	`DstrctNm`	`Max35Text`
Country subdivision	`CtrySubDvsn`	`Max35Text`
Country	`Ctry`	`CountryCode`, `[A-Z]{2,2}`
Address line	`AdrLine`	`Max70Text`, up to 7 occurrences

Three population patterns are possible and the industry has given them names. **Unstructured** uses `AdrLine` only. **Fully structured** uses the named elements and no `AdrLine` at all. **Hybrid** uses `AdrLine` for the street part but populates `TwnNm` and `Ctry` as discrete elements. CBPR+ narrows `AdrLine` to three occurrences; the Fedwire Funds Service hybrid profile requires at least country code and town name plus up to two free-format lines of seventy characters.

The deadline is firm and dated. Swift is removing the unstructured option with Standards Release 2026: from **November 2026**, a CBPR+ payment instruction carrying a purely unstructured postal address is rejected at the network, and only hybrid or fully structured addresses are accepted, with `TwnNm` and `Ctry` mandatory.

Compare what this replaced. MT 103 field 59 is `[/34x] 4*35x`: an optional account line followed by four lines of thirty-five characters holding name and address together, with no separator between the name and the address and no indication of which line is which. Field 50a option K is the same shape for the ordering customer. Every sanctions screening engine in the world has spent thirty years fuzzy-matching those blobs, and the false positive rate that follows is the reason a bank's payment investigations team is the size it is. Moving the town and the country into their own elements does not make screening perfect; it changes it from a text-similarity problem into a lookup, which is a different order of problem.

The reason this deadline slipped, repeatedly, has nothing to do with software. Banks cannot populate `TwnNm` for a customer whose stored address is four lines of free text typed into a branch terminal in 1998. Address remediation across tens of millions of customer records, with no reliable way to distinguish a street from a district in a language nobody in the remediation team reads, is a multi-year data project. Swift's own response has been to offer a machine learning service that infers structure from legacy address strings, which tells you how tractable the manual approach was judged to be.

Remittance information, exactly

`RmtInf` is typed `RemittanceInformation16` and has exactly two children:

- `Ustrd`, `Max140Text`, unbounded in the base schema.
- `Strd`, `StructuredRemittanceInformation16`, unbounded in the base schema.

`StructuredRemittanceInformation16` contains, in order: `RfrdDocInf` (referred document information, unbounded, each with `Tp`, `Nb` as `Max35Text`, `RltdDt` as `ISODate` and `LineDtls` for line-item detail), `RfrdDocAmt`, `CdtrRefInf` (with `Tp` and `Ref` as `Max35Text`), `Invcr`, `Invcee`, `TaxRmt`, `GrnshmtRmt`, and up to three occurrences of `AddtlRmtInf` as `Max140Text`.

That is where the three invoices in the worked example live: three `Strd` occurrences, or one `Strd` with three `RfrdDocInf` occurrences, each carrying its own number, date and amount. There is also `RltdRmtInf`, typed `RemittanceLocation7`, up to ten occurrences, which does not carry the remittance data at all but a pointer to it: a `Mtd` from `{FAXI, EDIC, URID, EMAL, POST, SMSM}` and an `ElctrncAdr` of up to 2,048 characters. That is the escape valve for genuinely large remittance sets, alongside the standalone `remt.001` remittance advice message.

Now the honest part, which vendor material tends to omit. CBPR+ restricts `Ustrd` to a single occurrence of 140 characters. MT 103 field 70 is `4*35x` — also 140 characters. For an institution that migrated by mapping field 70 into `Ustrd` and declaring victory, the cross-border remittance capacity did not increase by one byte. Every gain in this area comes from `Strd`, and `Strd` requires the originating corporate to supply invoice-level data, the originating bank to carry it, every intermediary to pass it unchanged, and the beneficiary's bank to present it on the statement. That is four consenting parties, and the Bank of England's decision in July 2026 to abandon its November 2027 structured remittance mandate for CHAPS is the clearest available evidence of how many of them were ready.

Codes that live outside the schema

A detail with disproportionate operational weight: many of the code sets are not in the schema. `ExternalPurpose1Code` is declared in the XSD merely as a string of one to four characters. `ExternalCategoryPurpose1Code` and `ExternalServiceLevel1Code` are the same; `ExternalLocalInstrument1Code` allows up to thirty-five. The permitted values live in the **External Code Sets**, published by the Registration Authority on a quarterly cycle at the end of February, May, August and November, in XLSX, XSD and JSON. The point of externalising them is that a new code can be added without incrementing a message version.

The consequence for an implementer is that validating a message against the XSD does not validate its codes, and that any implementation which compiles the code list into an enumeration in source will be wrong within three months. Treat the external code sets as reference data with a refresh job, exactly as you would treat a BIN table.

Purpose codes are the clearest example of what this buys. The Bank of England's recommended UK purpose code list includes, among others:

Code	Meaning
`PCOM`	Final payment to complete the purchase of a property
`PLDS`	Payment of funds from a lender as part of the issuance of a property loan
`PLRF`	Transfer or extension of a property financing arrangement to a new deal
`HLRP`	Property loan repayment
`HLST`	Property loan settlement
`INBC`	Settlement of interbank charges not being recharged to the debtor
`BKFE`	Bank loan fees, including agent, amendment and commitment fees

Code	Meaning
`CORT`	Trade settlement payment

Since **May 2025** the Bank has mandated purpose codes on all CHAPS payments between financial institutions and on property transactions, and Legal Entity Identifiers on all CHAPS payments between financial institutions. From **November 2027** purpose codes extend to all CHAPS payments, and LEIs to `pacs.004` returns and `pacs.009` COV payments between financial institutions. A four-character code declaring that a given £340,000 movement is a property completion rather than an unexplained transfer is worth more to a financial crime team than any amount of free text, and it costs eleven bytes.

Migration state, as of August 2026

Infrastructure	ISO 20022 date	Approach
SEPA (EPC schemes)	Native since scheme launch	Not applicable
T2, Eurosystem RTGS, replacing TARGET2	Live 20 March 2023, migration weekend 17 to 20 March	Big bang
EURO1, EBA Clearing	March 2023, aligned with T2	Big bang
Swift CBPR+, cross-border	Coexistence opened March 2023, closed 22 November 2025	Coexistence
CHAPS, Bank of England	June 2023, enhanced data mandated in phases from May 2025	Like-for-like, then enrichment
Lynx, Payments Canada	Introduced March 2023, MT retired 22 November 2025	Coexistence
CHIPS, The Clearing House	8 April 2024	Big bang
Fedwire Funds Service	14 July 2025	Big bang
Bacs	Not migrated; Standard 18 remains	Translation guide published

The pattern is not accidental. A domestic real-time gross settlement system has a closed participant list, a single operator, a mandate and a testing regime, so it can cut over in a weekend. T2 settled around 400,000 transactions on its first day, in line with the pre-migration average for a system that had been handling roughly €2.2 trillion a day. CHIPS processed 555,345 payments worth $1.81 trillion on its first day, 8 April 2024, figures its operator described as in line with a typical day. Fedwire, which averages more than $4.7 trillion a day, went in a single cutover on 14 July 2025. An open correspondent network with eleven thousand institutions cannot do that, which is why Swift ran two and a half years of coexistence and still needed contingency arrangements at the end.

Adoption figures track the deadline rather than the benefit. In June 2025 Swift reported that over 40% of daily traffic was in ISO 20022, more than 1.8 million payment messages a day across 180-plus sending and 220-plus receiving countries, and over 75% of its payments market infrastructure traffic. By September 2025 average daily ISO 20022 payment instructions had passed 60%; by October 2025 thirty-two RTGS market infrastructures had adopted the standard, representing 84.6% of Swift's global PMI traffic volume, and Lynx reported that over 98% of its messages were MX a month before the deadline. Read that curve honestly: the industry moved when the date got close, not when the business case was explained.

The forward calendar, again as of August 2026:

- **1 January 2026.** Swift's contingency conversion of MT messages sent by institutions that had not migrated, and its in-flow translation of inbound ISO 20022 into MT for receivers, both became chargeable.
- **November 2026, Standards Release 2026.** The unstructured postal address option is removed and rejected when used. MT 101 (single) moves to `pain.001` with contingency conversion available; MT 101 (multiple) reaches end of life. Receipt of `camt.110` via Case Management becomes mandatory.
- **November 2027.** Mandatory send and receive of `camt.110` and `camt.111` for exceptions and investigations, replacing in-flow translation. Payment cancellation moves exclusively to Case Management over FINplus. MT 199 and MT 299 for exceptions, investigations and payment initiation migrate. CHAPS extends purpose codes to all payments and LEIs to `pacs.004` and `pacs.009` COV.
- **November 2028.** Target for direct debits, charges and statement messages, with enforcement dependent on community adoption. Cheque messages continue indefinitely.

Bacs, and what a 100-character record cannot hold

The United Kingdom's bulk clearing has not migrated and is a useful antidote to the idea that ISO 20022 is universal. Bacs Standard 18 remains a fixed-length record:

Field	Position	Length	Content
Destination sorting code	1 to 6	6	Numeric
Destination account number	7 to 14	8	Numeric
Destination account type	15	1	Zero filled
Transaction code	16 to 17	2	`99`, `Z4`, `Z5` credits; `01`, `17`, `18`, `19` debits; `0N`, `0C`, `0S` AUDDIS
Originating sorting code	18 to 23	6	Numeric
Originating account number	24 to 31	8	Numeric
Free format	32 to 35	4	Blank, or `/` plus three characters
Amount in pence	36 to 46	11	Right justified, zero filled
Service user's name	47 to 64	18	Left justified
Service user's reference	65 to 82	18	Left justified
Destination account name	83 to 100	18	Left justified
Bacs processing day	101 to 106	6	`bYYDDD`, multi-day files only

One hundred characters, or one hundred and six for a multi-processing-day file. Bacs publishes an ISO 20022 to Standard 18 translation guide mapping `pain.001` to Direct Credits, `pain.008` to Direct Debits, `pacs.008` to cleared credits, `pacs.003` to cleared debits, `pacs.004` to returns, `pacs.007` to reversals, `pain.009` to AUDDIS, `pain.010` and `pain.011` to ADDACS, and `acmt.022` to AWACS, with dates converted from `YYYY-MM-DD` to the Bacs Julian form `bYYDDD` and lowercase folded to uppercase.

Look at the arithmetic and the limits of translation become obvious. The service user's reference is eighteen characters. A UETR is thirty-six. An `EndToEndId` is up to thirty-five. A `RemittanceInforma-`

`tion` block with three structured invoices is several hundred bytes. None of them fits, and no amount of implementation quality changes that. Translating a `pain.001` into Standard 18 is lossy by construction, and anyone building a translation layer should be able to enumerate, in writing, exactly which elements are dropped and hand that list to whoever signs off the migration.

Why a migration of this kind takes a decade

It is worth setting out the reasons as engineering rather than as complaint, because they generalise to every large format migration.

The value is bilateral but the cost is unilateral. Structured remittance data only pays when the sender populates it *and* the receiver consumes it. The first mover pays full cost for zero benefit. In that situation nothing happens until somebody imposes a date, which is why every milestone in the table above is a regulatory or operator mandate rather than a market outcome.

The message is the smallest part of the change. Behind a `pacs.008` sit sanctions screening, fraud scoring, liquidity and intraday position management, statement generation, reconciliation, archive, regulatory reporting and dispute handling — most of it written against thirty-five-character references and four-line addresses, some of it in COBOL, and much of it maintained by people who have retired. Changing the format at the edge is a quarter's work. Changing what the institution can *do* with the data is a decade's.

You cannot populate a field with data you have never collected. The address mandate slipped twice for this reason and no other.

Coexistence is not a bridge, it is a tax. For the duration, every institution runs two message stacks, two test suites, two sets of operational runbooks and two support rotas, and reconciles between them. That cost is why coexistence must have an end date, and why the end date must eventually stop moving. Swift's decision to make contingency conversion and in-flow translation chargeable from 1 January 2026 is the same lever, applied with a price rather than a prohibition.

The target moves during the project. Standards Releases are annual. A programme that began against `pacs.008.001.08` and runs for four years will finish inside a repository that has moved several versions on, which is why market infrastructures pin versions and why the pinned version becomes the real standard.

The forcing function is legal, not technical. The reason structured party data is mandatory is that FATF Recommendation 16 and its national implementations require identifiable originator and beneficiary information to travel with a transfer. Format follows obligation.

Coexistence, translation, and what to build

The mechanics of coexistence deserve stating precisely, because "we translate at the boundary" hides an important design decision. Swift's approach centred on Transaction Manager, which holds a central copy of the transaction so that the rich data survives even when an individual leg is delivered in a reduced format, plus in-flow translation from ISO 20022 to MT for receivers who had not migrated and, after November 2025, contingency conversion in the other direction for senders who had not. The important property is that the central copy, not the delivered message, is the record.

That generalises into the single rule worth taking away from this chapter. **Keep the ISO 20022 message as the system of record and translate only at the last mile, never in the middle.** A translated message can be revalidated; it cannot be reconstituted. Once a fifteen-element postal address has been flattened into four lines of thirty-five characters, the information required to rebuild it does not exist anywhere

in the message, and an institution that stores the flattened version has destroyed the very thing the migration was for.

Four other things follow, for anyone building against this.

Model, do not map. Store the message structure natively — nested parties, typed amounts, repeating remittance — and derive legacy formats on the way out. Every implementation that flattens ISO 20022 into an existing internal record on receipt has to be rebuilt later at greater cost, and the rebuild always arrives attached to a regulatory deadline.

Never treat the XSD as the specification. The schema permits far more than any network will accept. CBPR+ fixes `NbOfTxs` to 1; the schema does not. The schema allows seven address lines; CBPR+ allows three and Fedwire's hybrid profile allows two. The usage guideline published on MyStandards is the contract, and the schema is only the outer bound of it. A message that passes `xmllint` and fails the network is the normal case, not an anomaly.

Refresh the external code sets on a schedule. Quarterly, at the end of February, May, August and November. If a purpose code or a status reason code is an enumeration in your source tree, you have a defect with a known expiry date.

And test the round trip. Take a fully populated `pacs.008` with three structured invoices and a fully structured address, translate it to MT, translate it back, and diff the two. The diff is the true measure of what your coexistence arrangement costs, expressed in the only units that matter: the fields that arrive empty at the other end. Print it. Show it to whoever is signing the migration off. It is a more useful document than any readiness report, and it is the one nobody produces.

32.98 Common wrong ideas

Wrong: ISO 20022 is a message format that replaces ISO 8583. **Right:** It is a modelling standard with a repository and a registry behind it, and what it replaced was SWIFT MT for cross-border payment instructions and reporting, not card authorisation traffic.

Wrong: Card messages are about to move across too. **Right:** The `caaa`, `cain`, `catp` and `casp` business areas exist and have made almost no impression on global acquiring volume; the `0100` will outlive everyone reading this.

Wrong: ISO 20022 means XML. **Right:** Part 4 generates XML Schema, Part 8 generates ASN.1, a JSON Schema generation recommendation was approved in June 2025 and Part 9 generalises syntax generation altogether, and nexo card messages are routinely carried as ASN.1.

Wrong: Migrating gives you longer remittance fields. **Right:** The cross-border market practice permits one occurrence of the unstructured field at one hundred and forty characters, exactly the size of the field it replaced, and every real gain comes from the structured block, which needs four consenting parties to populate, carry, pass and present it.

Wrong: A message that validates against the schema will be accepted by the network. **Right:** The schema is only the outer bound; the market practice fixes the number of transactions at one and narrows the permitted address lines, and the usage guideline is the contract.

Wrong: The schema defines the permitted code values. **Right:** Purpose, category purpose, service level and local instrument codes are declared as plain strings whose values live in the External Code Sets, republished quarterly, so a code list compiled into source is a defect with a known expiry date.

Wrong: Translating at the boundary is a safe way to coexist. **Right:** Translation preserves delivery and not meaning, it is not reversible, and an institution that keeps the flattened message as its record has destroyed exactly the originator and beneficiary information the standard existed to carry.

Wrong: A settlement amount can be parsed into a floating-point type like any other number. **Right:** It is a decimal with up to eighteen total digits and five fractional digits, and parsing it as binary floating point produces a one-penny reconciliation break six months later.

Wrong: Five fractional digits mean the schema will catch an impossible currency amount. **Right:** A value with five decimal places is a valid decimal and an invalid euro amount, because the constraint that the value must respect the currency's minor units lives in the usage guideline and not in the schema.

Wrong: The industry migrated because the business case was compelling. **Right:** Adoption tracked the deadline rather than the benefit, because the value of structured data is bilateral while the cost of producing it is unilateral, so every milestone was a regulatory or operator mandate.

32.99 Chapter summary in 20 lines

1. ISO 20022 is not a format but a method for producing formats, plus a dictionary of business concepts and a committee structure for deciding what goes into the dictionary.
2. What retired on 22 November 2025 was SWIFT MT for cross-border payment instructions and reporting, not ISO 8583, and the card authorisation message is untouched.
3. The old numbered form is small and fast precisely because nothing in it says what anything is, which works until the day a message must carry something the code book never anticipated.
4. The new form labels every value, which makes it fatter, makes it self-describing, and makes the same concept mean the same thing in every kind of message the standard produces.
5. That consistency is the actual product: every form is built out of a central dictionary rather than invented from scratch.
6. Translated into money, three invoices that once travelled as a one-hundred-and-forty-character note now travel as three structured entries that a ledger can match and close without a human.
7. The architecture has three layers — a business model in UML, a logical message assembled from the Data Dictionary, and a physical syntax generated mechanically — and confusing them is the source of most arguments about the standard.
8. Governance is a registry rather than a standards body in the usual sense, with a Registration Authority maintaining the repository, a Registration Management Group overseeing it and Standards Evaluation Groups validating new definitions.
9. A message identifier carries business area, functionality, variant and version, and because the version is embedded in the namespace, upgrading is a release rather than a configuration change.
10. Market infrastructures therefore pin versions and hold them for years, and the pinned version becomes the real standard.
11. Set beside ISO 8583 the differences become concrete: a six-digit trace number that wraps within the day becomes a UUID unique end to end, and a two-character outcome becomes a status plus a separate mandatory reason.
12. Structured addresses are the least glamorous part of the migration and by a distance the most expensive, because they turn sanctions screening from a text-similarity problem into a lookup.
13. That deadline slipped for a reason that has nothing to do with software: a bank cannot populate a town name for a customer whose stored address is four lines of free text typed into a branch terminal in 1998.

14. Richer data is a capacity rather than a delivery, and the Bank of England's decision in July 2026 to abandon its structured remittance mandate for CHAPS is the clearest available evidence of how few parties were ready.
15. Many code sets live outside the schema and are republished quarterly, so schema validation does not validate codes and compiled enumerations rot within three months.
16. Purpose codes show what the structure actually buys: four characters declaring that a movement is a property completion rather than an unexplained transfer are worth more to a financial crime team than any amount of free text.
17. Domestic real-time gross settlement systems cut over in a single weekend because they have a closed participant list, one operator and a mandate; an open correspondent network of eleven thousand institutions cannot.
18. Adoption curves therefore track deadlines and not benefits, because the value is bilateral while the cost is unilateral and nobody can afford to go first.
19. Coexistence is not a bridge but a tax — two message stacks, two test suites, two runbooks, two support rotas — which is why it must have an end date and why the end date eventually had a price attached to it.
20. Keep the ISO 20022 message as the system of record, translate only at the last mile, never treat the schema as the specification, and test the round trip, because the fields that arrive empty at the other end are the only honest measure of what the arrangement costs.

Sources: ISO 20022 Registration Authority — ISO 20022 standard parts 1 to 8 (2013) and ISO 20022-9:2026; message definition catalogue; business area code list (RA, June 2025); external code sets publication policy; RMG governance pages; ISO 20022 Technical Support Group, Generation of JSON Schema Draft 2020-12 for ISO 20022 (v2.0, approved 10 June 2025). Message schemas: pacs.008.001.09 and pacs.008.001.07 XML Schemas as generated by the ISO 20022 Standards Editor. Swift: ISO 20022 implementation FAQs; ISO 20022 for financial institutions; ISO 20022 in bytes, "One month to go"; "Momentum builds as industry advances ISO 20022 adoption" (23 June 2025); CBPR+ roadmap beyond November 2025 (v5, June 2026); Swift AI address structuring model FAQ; CBPR+ pacs.008.001.08 usage guideline. Bank of England: Mandating ISO 20022 enhanced data in CHAPS (policy statement, 2024) and Expanding mandatory ISO 20022 enhanced data in CHAPS from 2027 (policy statement, 2025, including the July 2026 structured remittance decision); UK recommended purpose code list. European Central Bank: "Successful launch of new T2 wholesale payment system" (21 March 2023). EBA Clearing: EURO1 ISO 20022 migration. The Clearing House: CHIPS network migrates to ISO 20022 (10 April 2024). Federal Reserve Financial Services: Fedwire Funds Service ISO 20022 migration announcement (15 July 2025), ISO 20022 format FAQs, and Fedwire Funds Service ISO 20022 Quick Reference Guide. Payments Canada: Lynx ISO 20022 adoption and RTR pacs.002.001.10 usage guideline. European Payments Council: SEPA Credit Transfer Inter-PSP Implementation Guidelines 2025 v1.0. Bacs: ISO 20022 to Standard 18 Translation Guide and the Standard 18 message implementation guide. Swift Standards MT November 2021 Message Reference Guide for MT 103 field formats.

CHAPTER 11

The Physical Network

33.0 What this chapter gives you

1. You will be able to name the five roles a card authorisation passes through, and say which two of them are scheme members and therefore the only parties that can be fined or hold a settlement position.
2. You will be able to account for the 790 milliseconds of a contactless payment segment by segment, and explain why the card network is about five per cent of it while the phone and the reader are almost two thirds.
3. You will be able to separate the parts of a latency budget that are physics from the parts that are software, and say why an issuer processor in Virginia costs a London transaction about seventy-five milliseconds that no amount of money buys back.
4. You will be able to explain why a payment HSM is a computer that refuses rather than a safe that hides, and why zeroising one makes every key the institution holds permanently unreadable.
5. You will be able to trace a PIN block through translation at the acquirer, the scheme and the issuer, and say how many times the clear PIN exists and where.
6. You will be able to explain why a timeout is a financial event rather than a performance event, and what the 0400 reversal is for.
7. You will be able to say why partial failure is more dangerous than total failure, using Visa Europe's ten hours and ten minutes on Friday 1 June 2018 as the worked example.
8. You will be able to answer a due diligence question about HSM certification by naming the validated module and its certificate rather than the appliance.
9. You will be able to state the three phases of PCI PIN Security Requirement 18-3 and their dates, and say whether an estate still shipping bare encrypted keys to terminals is compliant.
10. You will be able to explain why moving scheme endpoints into the public cloud improves one firm's resilience while concentrating everybody's risk, and name the regime that now oversees that concentration.

At twenty to nine on a Tuesday morning a woman buys a sausage roll and a coffee in a bakery on Kirkstall Road in Leeds. Four pounds twenty. She holds her phone against the reader, the reader beeps, a green tick appears, and she is out of the door before the receipt has finished printing. The whole thing takes about a second.

Everything in this volume so far has described what was in that second as *messages*. A 0100 authorisation request with a bitmap and a set of data elements; a 0110 response carrying an approval in DE39 and a six-character code in DE38. That description is true and it is the one you need in order to reason about the system. It is also, in a specific sense, weightless. Messages do not travel. Optical and

electrical signals travel, along glass, through switches, into racks in buildings with concrete walls and diesel tanks.

This chapter is about the buildings. It is about which company owns which machine, what happens inside the box that no engineer is permitted to open, where the second actually goes, and what a merchant sees when the whole apparatus stops working on a Friday afternoon.

The plain version

A chain of desks

Imagine that the bakery cannot approve the payment itself, so it has to ask. The question travels down a chain of desks, and each desk does one job and hands it on.

At the first desk is the card reader on the counter. Its job is to read the phone, work out that the shop wants £4.20, write all of that down on a slip of paper in a very compressed shorthand, and hand it on.

The second desk belongs to the company the bakery pays to handle its card money: a building full of computers somewhere in England. It checks that the slip is properly filled in, works out which card network the payment belongs to, translates the shorthand into the shorthand *that network* prefers, and passes it on.

The third desk is the card network itself. It has no opinion about whether this payment is a good idea. Its job is to look at the front of the card number, work out which bank in the world issued it, and route the slip there. It is a sorting office, not a decision maker.

The fourth desk belongs to the bank's card system, and this is where the decision happens. Is this really her phone? Does the cryptographic signature the phone produced check out? Has she got £4.20 available? Has anything about this purchase looked strange in the last five minutes?

The fifth desk is the bank's actual accounts system: the ledger with her name on it and a balance next to it. Then the answer comes back down the chain in reverse, and the reader beeps.

The stopwatch

Now put real numbers on it, because the numbers are the point. Every figure below is in thousandths of a second — milliseconds.

Step	Time
Phone held against the reader, chip does its cryptographic sum	500
Reader to the payments company's building	25
Payments company checks and translates the slip	10
Payments company to the card network	5
Card network sorts and routes	20
Card network to the bank's card system	10
Bank decides	120
Bank's card system back to the card network	10
Card network back	20
Card network to the payments company	5
Payments company to the reader	25
Reader shows the tick and beeps	40
Total	**790**

Just under eight tenths of a second, and look at how it is spent. Almost two thirds of it is the phone and the reader talking to each other before anything leaves the shop at all. The card network — the thing everybody pictures when they imagine "the payment system" — accounts for forty of those eight hundred milliseconds, or five per cent. The travelling itself, all six legs of it, is seventy milliseconds. The single biggest slice after the phone is the bank thinking. So when a shopkeeper says the card machine is slow, the card network is almost never what they are complaining about.

Why there is spare time in the budget

People in the industry talk about a payment needing to complete in about two seconds. Eight hundred milliseconds is well inside that. The remaining twelve hundred are not waste; they are margin, and the margin exists because of specific, nameable things that go wrong. The bank's computers might be busy. The shop's broadband might be having a bad minute and a packet might have to be sent twice. The bank might not be in Britain.

That last one matters more than you would think, and it is the one thing in the entire chain that money cannot fix. Light in a glass fibre travels at about two hundred thousand kilometres per second, slower than light in a vacuum because glass slows it down. Leeds to London and back is roughly three milliseconds of pure travel. London to Virginia and back, which is where a great many card systems physically live, is about seventy-five. If the bank's computers are in Virginia, seventy-five milliseconds of your budget is gone before anybody has done any work, and no amount of money buys it back, because you cannot buy a faster universe.

The locked box

There is one desk in the chain where the person does not do the work themselves.

When the bank checks the customer's PIN, or checks the cryptographic signature the chip produced, it needs secret keys. Those keys are the crown jewels: anyone holding them could forge cards for millions of customers. So the bank does not keep them in an ordinary computer. It keeps them in a metal box bolted into a rack in a locked cage inside a locked room.

The box works like this. You post a question through a slot: "here is an encrypted PIN and here is the customer's account number — is the PIN right?" The box does the sum inside itself and posts out a single word: yes or no. What it will never do, no matter how you ask, is post out the PIN. There is no command for that. The engineers who built it deliberately left it out.

And if anybody tries to open the box, drill it, freeze it, or lift it off the rack, it does not resist. It erases itself, and what the thief carries away is an expensive paperweight. That is the opposite of how people imagine security. The box is not strong. The box is *suicidal*.

Two of everything

Because all of this is a chain, and because a chain stops if any link stops, everybody in it builds two of everything. Two card readers behind the counter. Two internet connections into the payments company. Two data centres, in two towns, each big enough to carry the whole load on its own.

And here is the thing a twelve-year-old works out faster than an adult: the frightening failure is not one building burning down. If a building burns down, everyone knows, and the other building takes over. The frightening failure is one building going *half* wrong. Still answering the phone. Still saying it is fine. Still sending confused messages to the other building. Nobody is sure whether to switch over, because switching over is itself dangerous, and while everybody argues, the queue at the bakery gets longer.

That is not hypothetical. It is what happened to Visa in Britain on Friday 1 June 2018, and we will walk through it.

Where the plain version stops being true

The first correction is that the five desks are five jobs, not five companies, and often not five buildings. The chain above is a chain of *roles*, and roles do not map one-to-one onto companies. Adyen and Stripe are the gateway, the licensed acquirer and the acquiring processor in one legal and technical stack, so three of my five hops become function calls between processes in the same estate. Conversely, one apparent hop routinely conceals six: what I drew as "the payments company checks the slip" may be a gateway, a tokenisation vault, a risk engine, a switch, a scheme endpoint and a message-repository write, each with a separate failure mode. Do not count boxes. Count trust boundaries — the places where data crosses from one organisation's liability into another's — and count clocks, because every boundary has a timer attached to it.

The second correction is that "two seconds" is a convention, not a specification. No card scheme publishes a rule saying an authorisation must complete within two seconds. What exists instead is a nest of separate timers with separate owners expiring at different moments: the terminal's timer waiting on its acquirer, the acquirer's on the scheme, the scheme's on the issuer, and the entirely unregulated timer running behind the customer's eyes. These disagree on purpose, and the consequences of each expiring differ. When the innermost one expires you do not get a slow approval; you get a *different transaction* — a reversal, a stand-in decision made by somebody who is not your bank, or a decline. Timeouts in payments are financial events, not performance events.

The third correction is that the locked box is a computer that refuses, not a safe that hides. The security of a hardware security module is not principally about concealment; it is about a deliberately impoverished instruction set. The device will take an encrypted PIN block and re-encrypt it under a different key, because that operation is necessary; it will not decrypt one and hand you the result, because that command was never implemented. Related, and routinely misunderstood: most keys are not stored *inside* the box. A large issuer's millions of card keys live in an ordinary database, each encrypted under a master key, and only that master key lives inside the tamper boundary. The box is not a vault holding your keys; it is the only place where your keys can be turned back into usable form, and only during an operation whose output you are permitted to see. Erasing it does not destroy your keys. It makes every copy of them, everywhere, permanently unreadable, which is worse.

The fourth correction is that "we have two data centres" is a claim about capacity, and the thing that fails is the switching. Building a second site that can carry the full load is expensive but conceptually simple. Detecting that the first site is unwell, distinguishing "unwell" from "briefly busy", and committing to a cutover that will itself cause disruption, is neither. The industry's own six-nines language conceals this. Availability of 99.9999 per cent sounds like a promise; it is 31.6 seconds of downtime in a year. The Visa Europe outage described later ran for ten hours and ten minutes: roughly one thousand one hundred and fifty years of a six-nines budget consumed in one afternoon. Such a number states design intent about steady-state operation. It says nothing about the tail, and the tail is what puts queues in bakeries.

The technical version

The five roles, stated exactly

The card industry's role names are used loosely in conversation and precisely in contracts. Here they are precisely.

Role	What it is	Licensed by a scheme?	Physical artefact
Terminal estate	POI devices on merchant counters, unattended units, softPOS on phones	No, but devices are PCI PTS approved	The PIN entry device
Payment gateway	Accepts the merchant's transaction, formats it for an acquirer	No	Servers, usually cloud
Acquirer	Holds the merchant agreement, is the scheme member, bears settlement risk	Yes, principal member	A licence and a balance sheet
Acquiring processor	Runs the switch that speaks to the scheme	No, acts for the acquirer	Data centres, switch, HSM farm
Scheme network	Routes, applies scheme rules, calculates interchange, clears and settles	It *is* the scheme	Global data centres
Issuer processor	Authorises on the issuer's behalf: cryptogram, PIN, risk, limits	No, acts for the issuer	Data centres, HSM farm
Issuer	Holds the cardholder account, owns the credit decision	Yes, principal member	A licence and a balance sheet
Core banking	The ledger of record for the customer's account	No	Mainframe or modern equivalent

Two of these are legal facts and the rest are engineering facts. Only the acquirer and the issuer are scheme members; only they can be fined, sanctioned or expelled, and only they can hold settlement positions. Everyone else in the chain, however large, is somebody's agent. This matters when things break, because the party a merchant shouts at is usually the processor and the party that owes the merchant money is always the acquirer. In Britain the acquiring names are Worldpay, Barclaycard, Global Payments, Elavon, Adyen and Stripe; the issuing-processor names include Fiserv, FIS, Thredd, Marqeta and Enfuce; the core banking names include Temenos, Finastra, Oracle, TCS BaNCS, Infosys Finacle, Thought Machine and Mambu. Ownership of these brands changes every few years. Their positions in the chain do not.

Inside an acquiring processor

An acquiring processor's estate is more than a switch, and a due diligence questionnaire will ask about each part of it by name.

A terminal management system, which knows every deployed device by serial number, pushes configuration and application updates, and injects or rotates terminal keys. An estate of tens of thousands of terminals is an estate of tens of thousands of key holders.

The switch itself: the process that receives an inbound transaction, validates it, applies merchant-level rules, routes it to the right scheme or domestic network, converts between message dialects, and applies the acquirer's own timers. Chapter 24 established that ISO 8583 is a family rather than a format, and the switch is where that family becomes a problem: the acquirer's internal dialect, Visa's, Mastercard's and a domestic scheme's are all different, and the switch owns the translation table.

A store-and-forward facility for transactions the terminal completed offline, which must be delivered later without being duplicated. A capture and clearing subsystem, which accumulates approved authorisations into the day's batch and generates the clearing files described in Chapter 30 — historically Visa BASE II and Mastercard IPM formats. An HSM farm, discussed below.

And finally the endpoint: the connection to each scheme. This is not a single wire. A processor typically holds multiple endpoints per scheme across multiple sites, each with its own institution identifier, its own key set and its own independent sequence of system trace audit numbers. DE11 is only unique within an endpoint within a day, which is precisely why Chapter 28 had to introduce four different identifiers rather than one.

The scheme switch

Visa's authorisation system began as BASE I, built by Bank of America and introduced in 1973 as the first real-time electronic card authorisation system; the batch clearing and settlement counterpart was BASE II. Both were later subsumed into V.I.P., the VisaNet Integrated Payment system. Mastercard's equivalents were INAS for authorisation and INET for clearing, carried over Banknet, and the modern architecture retains the split between a dual-message system, where authorisation and clearing are separate messages, and a single-message system, where one message does both. That split determines whether a refund can be a reversal or must be a fresh credit, and it is why debit and credit behave differently on the same terminal.

The routing decision itself is unglamorous. The scheme maintains an account range table mapping leading digits of the PAN to an issuer endpoint, the switch looks up the range, and the message goes there. Everything interesting is bolted around that lookup: inline fraud scoring, tokenisation lookups, currency conversion, and stand-in processing when the issuer does not answer.

The scale is worth stating with sources, because it is routinely exaggerated. Visa's stated figures as of October 2025 are 322 billion transactions a year, more than $16 trillion of payments volume, more than 150 million merchant locations, 4.8 billion credentials, seven independent data centres, availability of 99.9999 per cent, and a network processing up to 83,000 transaction messages per second worldwide. Its network operations centre stress-test capacity exceeds 65,000 messages per second, the figure Visa's public fact sheets have carried since 2017.

Divide 322 billion by the number of seconds in a year and the average is about 10,200 transactions per second. Set that against a peak of 83,000 messages per second and you have the number that actually governs the engineering: a peak-to-mean ratio of roughly eight to one. Card networks are not sized for the average. They are sized for the Friday before Christmas, and they spend most of the year at an eighth of capacity, which is why redundancy in this industry is cheaper than it is elsewhere.

Visa's Ashburn, Virginia site, which the company describes publicly, occupies 44 acres, is entered by only about 75 cleared employees, has 18-inch concrete walls rated to 170 mph winds, holds 500 petabytes of stored data, and is backed by 28 megawatts of standby power with 100,000 gallons of fuel for five days of autonomy. Those numbers are the answer to "what does a payment network physically look like".

The issuer side, and why authorisation does not touch the ledger

The issuer processor performs, in order: message validation; cryptogram verification, meaning it recomputes the ARQC the chip produced in DE55 and compares; cardholder verification, meaning online PIN verification if a PIN block is present; risk and fraud scoring; and an availability check.

That last step is where most people's mental model is wrong. The authorisation does not debit the customer's account. It writes to an authorisation record — variously the shadow balance, the open-to-buy or the memo post — that reduces available funds without altering the ledger balance. The ledger moves later, when the clearing record arrives, which is the mechanism behind every "pending transaction" a customer has ever queried and behind the whole of Chapter 30.

Behind the issuer processor sits core banking, which in a large bank is very often exactly what its reputation suggests: COBOL on IBM z/OS, batch-oriented, with a nightly cycle during which the ledger is closed for updating. The newer platforms are built around continuous processing precisely to remove that window. Whether the window exists is the single biggest determinant of what a bank can promise about timing, and it is a property of the ledger, not of the card system in front of it.

Why key operations happen inside tamper-responsive hardware

The threat being defended against is specific. ISO 9564 requires that a PIN never exist in cleartext outside a secure cryptographic device, because a PIN is four digits: an attacker who obtains encrypted PIN blocks and any oracle that will decrypt them can brute-force the space trivially. Equally, an issuer's card master keys can generate valid cryptograms for every card in a portfolio. Neither secret can be permitted to exist in the addressable memory of a general-purpose computer, however well patched, because a general-purpose computer's whole design goal is that its memory is addressable.

Three terms are used loosely and mean different things. **Tamper-evident** means an attack leaves visible traces. **Tamper-resistant** means an attack is made difficult by physical construction. **Tamper-responsive** means the device actively detects an attack in progress and reacts, and in a payment HSM the reaction is zeroisation: erasure of the local master key, after which every key the institution holds becomes undecryptable ciphertext. The sensors typically cover case opening, drilling, temperature excursions in both directions, voltage manipulation and radiation.

The market is small: Thales's payShield line and Utimaco's Atalla and CryptoSec families cover most of it, alongside a handful of others. Take the current Thales device as a concrete specimen, since generalities are useless here. The payShield 10K, in its PS10-S, PS10-D and PS10-F variants, appears on the PCI Security Standards Council's approved devices list under approval number 4-40266, evaluated against version 3.x of the PCI HSM Security Requirements, with an approval expiry of 30 April 2028. Its listed approved usage is *Restricted*, meaning the approval holds only when the device is deployed in an environment meeting at least the security requirements of a Controlled Environment as defined in the PCI key management requirements and the device's own security policy. It supports ISO Format 4 (AES) PIN blocks and is listed as supporting Remote Administration.

Separately, and this is the distinction practitioners get wrong, its FIPS validation is not of the whole appliance. NIST certificate #3610 covers the Thales Advanced Security Platform, a multi-chip embedded module *inside* the payShield 10000 family, validated to FIPS 140-2 overall Level 3 by UL Verification Services in January 2020 and now moved to the historical list as part of the general sunsetting of FIPS 140-2. If a supplier questionnaire asks you whether your HSM is "FIPS 140-2 Level 3 certified", the honest answer names the module and the certificate, not the box.

Published specifications for the same device: Triple DES at 112 and 168 bits, AES at 128, 192 and 256, RSA up to 4096 bits, and the FIPS 186-3 curves P-256, P-384 and P-521; host connectivity over

TCP/IP and UDP on dual 1 Gbps or 10 Gbps Ethernet ports, with an optional single FICON port for mainframe attachment; and conformance claims against ISO 9564, 10118, 11568, 13491 and 16609, ANSI X3.92, X9.8, X9.9 and X9.17, ASC X9 TR-31 and X9.143, TR-34 and X9.139, and APACS 40 and 70. The presence of FICON on a 2020s product tells you what these devices are still plugged into.

The key hierarchy, named

Key	Full name	Where it lives	What it protects
LMK	Local master key	Inside the HSM tamper boundary only	Every other key the institution holds
ZMK / KEK	Zone master key, key encrypting key	Exchanged between two institutions	The transport of working keys
ZPK	Zone PIN key	Under LMK at rest, under ZMK in transit	PIN blocks between two parties
BDK	Base derivation key	Under LMK; never leaves the HSM	Derives per-terminal, per-transaction keys
PVK	PIN verification key	Under LMK	PIN verification values
CVK	Card verification key	Under LMK	CVV, CVC, iCVV, dCVV values
IMK	Issuer master key	Under LMK	Derives per-card keys for EMV
UDK	Unique DEA key	Derived per card	Generates and verifies the ARQC

Only the LMK is physically inside the device. Everything else is a database column holding ciphertext, meaningful only when handed back through the HSM's slot. That is what makes the architecture affordable, and what makes zeroisation catastrophic.

DUKPT, Derived Unique Key Per Transaction, standardised in ANSI X9.24, solves a specific problem: a terminal on a counter is physically accessible to the attacker. So the terminal is never given a long-lived key. It is injected with an initial key derived from the acquirer's base derivation key and its own device identifier, and derives a fresh key for every transaction, discarding the previous one irreversibly. Compromising a terminal therefore yields the transactions after the compromise and none before it, and nothing about any other terminal.

Key blocks, and a compliance date you must know

For decades, keys were shared between institutions as bare encrypted values with the key's *purpose* recorded separately, in documentation, by convention. This is an obvious flaw: an attacker who can persuade a system to use a PIN key as if it were a data key can extract secrets the design never intended to expose. Key blocks fix this by cryptographically binding a key's permitted usage to the key itself, so that the binding cannot be altered without invalidating the block. The relevant standards are ASC X9 TR-31 and its successor ANSI X9.143.

PCI PIN Security Requirement 18-3 mandates key blocks in three phases, and the dates are exact. Phase 1, internal connections and key storage within service provider environments including all applications and databases connected to an HSM, took effect on 1 June 2019. Phase 2, external connections to associations and networks, took effect on 1 January 2023. Phase 3, extending key blocks to all merchant hosts, point-of-sale devices and ATMs, took effect on 1 January 2025. Anyone still transporting bare encrypted keys to a terminal estate as of this writing is out of compliance and has been for over a year and a half.

PIN translation, hop by hop

This is the single most instructive walk in the chapter, because it shows the tamper boundary doing real work.

A customer enters a PIN on a PIN entry device. The device constructs a PIN block — a fixed-length structure combining the PIN with part of the PAN so that identical PINs on different cards do not produce identical blocks — and encrypts it under a key derived by DUKPT for that transaction only. The clear PIN exists in the PED's secure processor for a few milliseconds and is then gone.

The encrypted block travels to the acquiring processor. Its HSM is given the block, the key serial number and the target key, and performs a *translation*: internally it derives the terminal's transaction key, decrypts the block, re-encrypts it under the zone PIN key shared with the scheme, and outputs the new block. The clear PIN existed for microseconds, inside the tamper boundary, and never entered the memory of the switch. At the scheme the same operation happens again: decrypt under the acquirer's ZPK, re-encrypt under the issuer's ZPK. At the issuer processor the block is finally verified rather than translated, decrypted inside the HSM and checked against a PIN verification value computed with the PVK, or against an offset, with only a yes or no coming back out.

The PIN therefore crosses three organisations and exists in the clear four times, each time for microseconds, each time inside a tamper-responsive device that has no command capable of revealing it.

The block formats are specified in ISO 9564-1. Formats 0 to 3 are built for Triple DES's 64-bit block. Format 4, standardised in 2015, is built for AES's 128-bit block and, unlike its predecessors, includes randomness so that the same PIN, PAN and key produce a different block every time. PCI's device requirements now force the migration: POI devices at version 5 and above supporting online PIN, and HSMs at version 4 and above supporting PIN processing, are required to support ISO Format 4.

The latency budget, restated with owners

The plain version's table was honest but anonymous. Here it is again with the two questions a performance engineer asks: who owns this slice, and is it physics or software?

Segment	Typical	Owner	Nature
Card or phone in the contactless field	up to ~500 ms	EMVCo kernel, card, reader	Protocol and silicon
Terminal to acquirer over broadband or 4G	20–40 ms	Merchant's network, acquirer edge	Mostly software and access network
Acquirer switch processing	5–20 ms	Acquiring processor	Software
Acquirer to scheme endpoint	2–10 ms	Private circuit or Edge connection	Physics plus equipment
Scheme routing and inline scoring	10–40 ms	Scheme	Software
Scheme to issuer processor	2–80 ms	Depends entirely on geography	Physics
Issuer decision, including HSM calls	50–250 ms	Issuer processor and core banking	Software and database
Return path	mirrors outbound	as above	as above
Terminal display and receipt	30–80 ms	Terminal	Software and printer

The contactless figure is the one hard number in the table: the EMV contactless specifications set the requirement that a card need not remain in the reader's field for more than about 500 milliseconds. Everything else is a range observed in production, not a published limit, and it moves with load.

Two conclusions follow. In a domestic transaction the network is a rounding error and the issuer's database is the story, so performance work belongs in the issuer processor rather than the wires. In a cross-border transaction the network is not a rounding error at all, and no amount of software engineering will help, because the constraint is geometry.

What physics charges you

Light in a vacuum travels 299,792,458 metres per second. In single-mode fibre, with a refractive index of about 1.468, it travels roughly 204,200 kilometres per second. The rule of thumb used by optical engineers is 4.9 microseconds per kilometre, and it is accurate enough for any budget you will ever build.

Route	Approximate fibre path	Round-trip floor	Typically observed
Leeds to London	~330 km	~3.2 ms	8–15 ms
London to Frankfurt	~750 km	~7.4 ms	12–20 ms
London to Ashburn, Virginia	~6,500 km	~64 ms	70–80 ms
London to Singapore	~11,000 km	~108 ms	160–200 ms

The gap between the floor and the observed figure is equipment: optical amplifiers, regenerators, routers, firewalls and the serialisation delay of putting bits onto a wire. It is bounded, and it is the part you can buy your way out of. The floor is not.

There is a second physics-adjacent cost engineers forget. A TLS 1.2 full handshake costs two round trips before any application data flows; TLS 1.3 reduces that to one. On a London-to-Virginia link that is 150 milliseconds of handshake for TLS 1.2 and 75 for TLS 1.3, on top of the TCP handshake. This is why payment links are never opened per transaction. They are long-lived, pooled, persistent sessions, kept warm and monitored, and a transaction that has to open a fresh connection has already lost more time than the entire issuer decision would have taken.

The heartbeat, and what it is for

Keeping a session warm requires knowing whether it is alive, which is the purpose of the 8xx message class introduced in Chapter 24. A 0800 network management request with the appropriate function code is an echo test; the peer answers 0810. The same class carries sign-on, sign-off and key change. Endpoints exchange echoes on a fixed interval, and after a configured number of consecutive failures the link is marked down and traffic redistributed to the surviving endpoints.

That is why partial failure is so much more dangerous than total failure. A dead link fails its echoes and is removed automatically in seconds. A link that answers every echo correctly while mishandling the transactions behind it will never be removed by any automatic process, because by the only test the system applies, it is healthy.

Timers, reversals and stand-in

When the issuer does not answer within the scheme's response time parameters, something must happen, and what happens is defined rather than improvised.

The scheme may substitute its own decision. Both major schemes operate stand-in processing, in which the network authorises on the issuer's behalf against limits the issuer has pre-registered. Visa's published action code 91 covers the case where stand-in is *not* available: "Issuer or switch inoperative and STIP not applicable or not available for this transaction; time-out when no stand-in". Issuer processors document a related behaviour in which Visa forces single-request stand-in and converts a 91 or 96 into an N0. Chapter 29 covered what these codes mean to a merchant; the point here is that they are emitted by network conditions rather than by any bank's opinion of the cardholder.

When the acquirer's own timer expires, it must assume the issuer may have approved and reserved funds it does not know about, and must therefore send a reversal — the 0400 reversal request of Mastercard's Transaction Processing Rules and its equivalents. The reversal is what stops a network hiccup from silently holding a customer's money for a week, and unwinding un-reversed timeouts is a large share of the reconciliation work described in Volume V.

Redundancy in practice

Serious payment estates use one of two patterns. **Active-active** runs both sites live, splitting traffic, so a site loss is a capacity loss rather than an outage and the failover has been exercised continuously. **Active-passive** runs one site live and one warm, which is cheaper and has the fatal property that the failover path is exercised only in the emergency.

Underneath sits the replication decision. **Synchronous** replication does not acknowledge a write until the second site has it, giving a recovery point objective of zero at the cost of adding the inter-site round trip to every write. That is why paired sites sit tens of kilometres apart rather than hundreds: at 4.9 microseconds per kilometre, a 50 km separation costs about half a millisecond per write and a 500 km separation costs five. **Asynchronous** replication removes that cost and accepts a non-zero recovery point objective. In payments that window is not an abstraction; it is a set of authorisations that happened, that customers were told about, and that the surviving site has never heard of.

The genuinely hard problem is neither of these. It is the decision. A cutover is disruptive in itself, so nobody wants to trigger one on a false positive; but the detection signal in a partial failure is ambiguous by definition. Firms therefore write runbooks with explicit trigger thresholds and rehearse them. Since 31 March 2025, when the transition period for the FCA's PS21/3 operational resilience regime ended, UK firms have been required to have identified their important business services, set impact tolerances for each, mapped the resources supporting them, and demonstrated by testing that they can stay within those tolerances in severe but plausible scenarios.

Friday 1 June 2018, narrated

The best public account of a scheme outage in Britain is Visa Europe's own, given to the House of Commons Treasury Committee by its chief executive Charlotte Hogg in a letter dated 15 June 2018. What follows is from that account and the contemporaneous reporting of it.

Visa Europe operated two data centres in the United Kingdom, either of which could independently handle 100 per cent of Visa's European transactions. In normal operation the systems were synchronised and either centre could take over from the other immediately.

At 14:35 on Friday 1 June 2018, a component within a switch in the primary data centre suffered what Hogg described as a very rare, partial failure. Because the failure was partial, the backup switch did not activate.

This is the exact scenario the seam warned about. The malfunctioning system at the primary site did not stop. It continued attempting to synchronise messages with the secondary site, building a backlog there which in turn degraded the secondary's own ability to process incoming transactions. The redundant capacity was not merely unused; it was being poisoned by the failing primary. Isolating the primary took nearly five hours, and full correct processing was not restored until 00:45 the following morning: ten hours and ten minutes end to end.

The numbers Visa gave Parliament are worth recording precisely. Across Europe, 51.2 million transactions were attempted during the affected period; about 10 per cent were affected, and 5.2 million failed to process. In the United Kingdom 2.4 million failed, affecting 1.7 million credit and debit cards; a further 2.8 million failed elsewhere in Europe. Disruption was not constant. There were two peaks — roughly ten minutes just after 15:00, and fifty minutes between 17:40 and 18:30 — during which 35 per cent of transactions failed. Outside those peaks the UK failure rate was closer to 7 per cent.

That last detail is the one merchants remember, because it describes what an outage actually feels like on a shop floor. Not "the card machines are down". Instead: one payment in fourteen fails for no visible reason, then briefly one in three, then one in fourteen again, with no announcement, no error message worth reading, and a customer at the front of the queue re-presenting the same card three times until it works. Staff cannot tell a scheme outage from a bad card, because on the counter they look identical. Visa engaged EY to conduct an independent review and said it would migrate European processing onto its global VisaNet platform, which it described as more resilient in detecting and recovering from partial malfunctions of exactly that type.

Three lessons generalise. Partial failure is worse than total failure. A hot standby coupled to a sick primary is a liability rather than an asset, because the coupling is the failure path. And the recovery clock is dominated not by repair but by diagnosis and by the decision to isolate.

From leased lines to TLS over the public internet

The connectivity story runs in four eras and the last one is still arriving.

Dedicated circuits, 1973 to the mid-1990s. BASE I ran over leased lines between member banks and Visa's centres. Terminals dialled: a countertop unit picked up a phone line, dialled a modem pool and took fifteen to thirty seconds, which is why floor limits and offline authorisation, covered in Volume II, were not an optimisation but a necessity. Behind the scenes ran X.25, and in bank environments IBM's SNA and SDLC.

Private packet networks, the 1990s and 2000s. Frame relay and then MPLS replaced point-to-point circuits with managed private clouds, and the scheme endpoint became a physical appliance installed in the customer's data centre — a Visa access point, or Mastercard's Interface Processor, the MIP. The scheme shipped you a box, and the box was the border.

IP with cryptographic protection, the 2010s. Terminals moved to ADSL, then Ethernet, then 4G, and the transport became the ordinary internet with TLS carrying the security a private circuit had previously carried by being private. PCI DSS Requirement 4 governs this directly: strong cryptography must safeguard the primary account number during transmission over open, public networks, with TLS 1.2 or higher the practical floor. Version 4 added Requirement 4.2.1, that certificates used to protect PAN in transit are confirmed valid and not expired or revoked, and 4.2.1.1, that an inventory of those

trusted keys and certificates is maintained. The second catches firms out, because certificates expire on a schedule nobody is monitoring until the morning they do.

Cloud and colocation endpoints, now. The physical box in the customer's data centre is being retired. Mastercard publishes three connectivity flavours side by side: Mastercard Edge for customers in their own on-premises data centres, Co-Lo Edge for those in third-party colocation facilities, and Cloud Edge for those in the public cloud, delivered in collaboration with cloud providers including AWS. A processor that runs no data centre of its own can now hold a scheme endpoint, which was not true a decade ago. Merchant-side, the same arc ends in softPOS: a commodity phone acting as the contactless reader, with no dedicated hardware between the card and the internet at all.

Concentration, and the critical third parties regime

Moving payment infrastructure to a small number of cloud providers improves the resilience of any single firm and concentrates the risk of all of them.

On 20 October 2025 that argument stopped being theoretical. Beginning at approximately 06:48 UTC, a latent race condition in DynamoDB's automated DNS management in Amazon Web Services' us-east-1 region produced an empty DNS record for the regional DynamoDB endpoint, which the automation could not repair. DNS resolution was restored by 09:40 UTC, but the cascade through dependent services — around 140 AWS services, including EC2 — meant full recovery took over fifteen hours. Firms whose control planes lived in that region discovered that their own multi-availability-zone architecture had not made them independent of it.

Regulators had already moved. The Critical Third Parties regime was created by the Financial Services and Markets Act 2023, and its rules were finalised jointly by the Bank of England, the PRA and the FCA on 12 November 2024, published as PS16/24 by the Bank and PRA and PS24/16 by the FCA. Designation is HM Treasury's decision, not the regulators'. On 10 July 2026 the Treasury made the first designations, effective 13 July 2026: Microsoft Ireland Operations Limited, Google Cloud EMEA Limited, Amazon Web Services EMEA SARL and Oracle Corporation UK Limited. Oversight applies only to the systemic services those firms provide to the financial sector, and the regime does not transfer responsibility: firms remain accountable for managing the risks in their own third-party arrangements. In the European Union the parallel instrument is the Digital Operational Resilience Act, which has applied since 17 January 2025.

Design rules that follow

Count trust boundaries, not companies. Every boundary has a timer, a key relationship and a different party to telephone at three in the morning, and your runbook needs all three written down before you need them.

Budget latency by segment and label each segment physics or software. You can optimise software; you cannot optimise distance, so decide where the issuer processor physically sits before you write any code. And never open a TLS session per transaction: pool, persist, keep warm and echo.

Treat a timeout as a financial event. Every one must produce a reversal or a reconciliation exception, and an exception nobody works is a customer complaint with a delay fuse.

Assume partial failure. Health checks must exercise the actual work rather than answer a ping, because a component that passes its heartbeat while corrupting its output will never be evicted by any automatic mechanism. Then rehearse the cutover, and rehearse the decision to cut over, separately. The technical failover is the easy half.

Keep the key inventory and the certificate inventory as first-class operational assets, and when you make a compliance claim about an HSM, name the validated module and its certificate rather than the appliance. On a due diligence questionnaire, precision reads as competence.

33.98 Common wrong ideas

Wrong: the five desks are five companies in five buildings. **Right:** they are five roles, and Adyen or Stripe may hold three of them in one stack while a single apparent hop conceals a gateway, a token vault, a risk engine, a switch, a scheme endpoint and a repository write.

Wrong: the card network is what makes a card machine slow. **Right:** the scheme accounts for about forty of the eight hundred milliseconds; the phone-to-reader exchange and the issuer's decision are where the time actually goes.

Wrong: the schemes publish a two-second rule for authorisation. **Right:** no such specification exists, only a nest of separate timers with separate owners — terminal, acquirer, scheme, issuer and the customer — expiring at different moments with different consequences.

Wrong: a timeout produces a slow approval. **Right:** it produces a different transaction: a reversal, a stand-in decision made by somebody who is not your bank, or a decline.

Wrong: an HSM is a safe that hides your keys. **Right:** almost every key lives as ciphertext in an ordinary database and only the local master key sits inside the tamper boundary; the security comes from a deliberately impoverished instruction set with no command capable of revealing a PIN.

Wrong: zeroisation protects your keys. **Right:** it makes every copy of them, everywhere, permanently unreadable, which is worse than losing the box.

Wrong: having two data centres means you have failover. **Right:** capacity is the easy half; detecting that a site is unwell, distinguishing unwell from briefly busy, and committing to a cutover that is itself disruptive is the part that fails.

Wrong: six-nines availability means outages are negligible. **Right:** 99.9999 per cent is 31.6 seconds a year, and the Visa Europe outage consumed roughly one thousand one hundred and fifty years of that budget in a single afternoon.

Wrong: authorisation debits the customer's account. **Right:** it writes an authorisation record — shadow balance, open-to-buy or memo post — that reduces available funds, and the ledger moves later when the clearing record arrives.

Wrong: "our HSM is FIPS 140-2 Level 3 certified" describes the appliance. **Right:** NIST certificate #3610 covers the Thales Advanced Security Platform, a module inside the payShield 10000 family, and an honest answer names the module and the certificate rather than the box.

33.99 Chapter summary in 20 lines

1. This book has described a payment as messages, but messages do not travel; optical and electrical signals travel, through glass and switches into racks in buildings with concrete walls and diesel tanks.
2. An authorisation passes through five jobs — the terminal, the merchant's payments company, the scheme network, the issuer's card system and the bank's ledger — and the answer returns down the same chain.

3. Timed end to end, a typical contactless approval takes about 790 milliseconds, comfortably inside the two seconds the industry talks about.
4. Almost two thirds of that is the phone and the reader talking before anything leaves the shop, the scheme is about five per cent, and all six network legs together are seventy milliseconds.
5. The remaining margin exists for busy issuers, retransmitted packets and geography, and geography is the only one money cannot fix.
6. Light in single-mode fibre travels about 204,200 kilometres per second, roughly 4.9 microseconds per kilometre, so London to Ashburn and back has a floor near sixty-four milliseconds however much you spend.
7. Because a full TLS handshake costs one or two round trips on top of that, payment links are long-lived, pooled and kept warm rather than opened per transaction.
8. The five desks are roles rather than companies, so the honest unit of analysis is the trust boundary, and every boundary carries a timer, a key relationship and a different party to telephone at three in the morning.
9. Only the acquirer and the issuer are scheme members, which is why the party a merchant shouts at is usually the processor while the party that owes the merchant money is always the acquirer.
10. The scheme switch itself is unglamorous — an account range table maps the leading digits of the card number to an issuer endpoint — and fraud scoring, tokenisation, currency conversion and stand-in are bolted around that lookup.
11. Visa's published 322 billion transactions a year against a peak of 83,000 messages a second implies a peak-to-mean ratio of about eight to one, which is why these networks are sized for the Friday before Christmas and why redundancy is comparatively cheap.
12. On the issuer side, authorisation verifies the chip's cryptogram, the PIN and the risk position and then writes a shadow balance rather than touching the ledger, which is what every pending transaction really is.
13. A PIN is four digits and an issuer master key can forge a whole portfolio, so neither may exist in the addressable memory of a general-purpose computer, and key operations happen inside tamper-responsive hardware that erases itself when attacked.
14. Only the local master key lives inside that hardware; everything else is a database column of ciphertext, which is what makes the architecture affordable and zeroisation catastrophic.
15. DUKPT gives each terminal a key that changes every transaction, so compromising a counter-top device yields nothing before the compromise and nothing about any other device.
16. Key blocks bind a key's permitted usage to the key itself, and PCI PIN Security Requirement 18-3 has extended that obligation to merchant hosts, point-of-sale devices and ATMs since 1 January 2025.
17. A PIN block is translated rather than revealed at each hop, existing in the clear for microseconds inside three organisations' tamper boundaries and never entering the memory of any switch.
18. Redundancy fails at the decision rather than the capacity: an 8xx echo evicts a dead link in seconds, but a link that answers every heartbeat correctly while mishandling the traffic behind it will never be evicted by any automatic mechanism.
19. That is exactly what happened to Visa Europe on 1 June 2018, when a partial switch failure poisoned the healthy site, took nearly five hours to isolate, and left 5.2 million European transactions unprocessed with UK failure rates oscillating between 7 and 35 per cent.
20. The rules that follow are to count trust boundaries rather than companies, to label every latency segment physics or software, to treat every timeout as a financial event that must produce a reversal, to health-check the real work rather than a ping, and to rehearse the decision to cut over separately from the cutover itself.

Sources: Visa corporate publications on VisaNet and the Ashburn data centre (October 2025) and Visa Europe fact sheet; Visa Europe's letter of 15 June 2018 to the Treasury Committee, with reporting by Finextra and The Guardian; Visa Developer response code documentation; Mastercard Edge Connectivity and Transaction Processing Rules; PCI SSC bulletin on PIN Security Requirement 18-3 and the PTS approved devices listing for approval 4-40266; NIST CMVP certificate #3610; Thales payShield 10K and Utimaco payment HSM specifications; EMVCo contactless specifications; FCA PS21/3 and the joint Bank of England, PRA and FCA policy statements PS16/24 and PS24/16; HM Treasury's designation announcement of 10 July 2026; AWS post-event summary and ThousandEyes analysis of the us-east-1 incident of 20 October 2025.

ABOUT THE AUTHOR

Shikhar Singh

Founder & Chairman

KedByte Technologies Private Limited

Shikhar Singh is the Founder & Chairman of KedByte Technologies Private Limited.

He wrote this book from the position of someone building payment software rather than someone commentating on it, and the difference shows in what it chooses to explain. The chapters that go deepest are the ones where he has watched capable engineers lose days: the gap between authorisation and settlement, the several identifiers all called "the transaction number", and reconciliation, which is unglamorous, unavoidable, and where more young financial technology companies fail than for any more interesting reason.

His conviction is that payments has no genuinely hard idea in it, only a very long chain of simple ones, most of which are documented for people who already understand them. The barrier is not difficulty. It is that the industry writes for itself. That is a solvable problem, and this book is an attempt at solving it.

Under his direction, KedByte Technologies Private Limited treats technical education as infrastructure rather than as a product feature: knowledge that should be transferable, verifiable, and free of the vocabulary barriers that keep capable people out of the field.

ABOUT THE PUBLISHER

KedByte Technologies Private Limited

KedByte Technologies Private Limited publishes technical work under the KedByte Press imprint.

The editorial standard applied to this volume is simple to state and difficult to meet. Every explanation must work twice: once for a reader who has never opened a terminal, and once for an engineer who needs the exact figure. Every claim that can be checked must be checked. Every figure that will go stale must carry a date. Where the field genuinely disagrees, both positions are given rather than one being quietly chosen.

Where this book uses real evidence, it uses it unaltered. The network trace in Part F, the failed pushes in Part G and the error messages quoted throughout are reproduced exactly as they were observed. Diagnosis is taught against ambiguous real data, because that is the only kind there is.

KEDBYTE

TECHNOLOGIES PRIVATE LIMITED

COLOPHON

This first edition of *How Money Moves* runs to five volumes and fifty-four chapters, and was set entirely from plain text.

The body text is set in TeX Gyre Pagella, a digital revival of Hermann Zapf's Palatino, chosen for long-form reading. Headings, tables and the KedByte identity are set in Poppins. Code, command output and diagrams are set in DejaVu Sans Mono, and every diagram in this book is drawn in plain ASCII so that it survives being copied anywhere.

The book was composed with pandoc and typeset with XeLaTeX on A4 stock. The single accent colour used on rules, chapter numbers and section headings is KedByte navy.

Shikhar Singh

Founder & Chairman, KedByte Technologies Private Limited

First Edition • KedByte Press