The Conserved Pipeline: Stage Gates, Pending Inventory, and Locating the Constraint in an Order-to-Cash System

Architecture, Computational Methods, and Reported Outcomes from MileSoft Manufacturing ERP

MileSoft Engineering Research Group

MileSoft Software Technologies

https://milesoft.net/

Corresponding authors: *Deepak Daryani* (deepakdaryani@milesoft.net), *Tanuj Daryani* (tanujdaryani@milesoft.net)

Working paper — Version 1.0 — 23 August 2026

Permanent link: https://milesoft.net/research/products/erp

Abstract

An order is not a record. It is an object that moves through a fixed sequence of gates — sold, planned, produced, delivered, invoiced — and at every gate it either passes or waits. The set of orders waiting at a gate is inventory in the strict sense, and it obeys the same conservation and flow relations as material inventory does.

This paper presents the architecture and computational methods of MileSoft Manufacturing ERP through that lens. We show that the product's own dashboard reports a conserved cascade: at each gate, the orders cleared plus the orders pending sum exactly to the count cleared at the gate before it. That identity is not a display convention — it is the property that makes stage conversion, per-gate work in process and constraint location computable from five numbers.

Four results follow. Constraint location reduces to finding the gate holding the largest pending set, and the observed capture puts that unambiguously at delivery rather than production. Stage flow time is Little's Law applied per gate, which converts a pending count into days of delay without any additional instrumentation. Order-to-cash does not end at the invoice gate: payment terms observed on the sales order extend the cycle by end of month plus ninety days, so the pipeline the dashboard shows is roughly half of the one that matters to the business. And monetary quantities in such a pipeline must be computed in exact decimal arithmetic against ISO 4217 minor units, because binary floating point does not reconcile a line to an invoice.

The paper is anchored in IEC 62264 for the enterprise-to-control boundary, Orlicky for the requirements-planning lineage, ISO 22400 for the KPI discipline, and Incoterms 2020 for the delivery terms an order header carries. A four-module deployment in which this product was one of four reported a phased rollout of

eight and six weeks per plant; that figure belongs to the programme and is not attributed to this module alone.

Keywords: Manufacturing ERP software; Order to cash pipeline; Production sheet; Sales order to invoice; Stage gate conversion; Work in process orders; Constraint identification; ISA-95 level 3 level 4 boundary; Material requirements planning; Bill of materials versioning; Engineering change order; Days sales outstanding

1 Introduction

The familiar manufacturing ERP story is a fourteen-month implementation of a brand-name system, a go-live full of workarounds, and a shop floor that quietly keeps running on the spreadsheets the ERP was meant to replace.

The product's own material gives the reason: most enterprise systems were built outward-in, general ledger first and a thin manufacturing module last, which is the wrong order for a factory. That is a good diagnosis and this paper takes it as read. What it adds is a way to measure whether a given system is actually running the floor, using data such a system already displays.

The measurement rests on one observation. An order is an object that passes through a fixed sequence of gates, and at each gate it either clears or waits. The waiting orders are inventory, and inventory obeys conservation. Everything in Sections 4 and 6 follows from that.

1.1 The eleven o'clock test

The product's material proposes a test worth repeating: ask a shop-floor supervisor when their team sees a schedule change made at eleven in the morning. If the answer is tomorrow's stand-up, the system is generating reports rather than running the floor.

It is a good test because it is unfalsifiable by a specification sheet and answerable in one question. This paper generalises it. A pipeline whose gate counts are visible and conserved can be interrogated the same way at every stage, and Section 6.3 turns that into eight such questions.

> The distinction is not real time against batch. It is whether the number a decision-maker sees is the number the floor is working from — and a system can fail that with a millisecond refresh if the two are different objects.

1.2 Contributions

1. The order pipeline stated as a conserved cascade, with the identity verified against the product's own reported counts (Section 4.1).
2. Constraint location from pending counts alone, requiring no cycle-time study (Section 4.2).
3. Stage flow time as Little's Law per gate, converting a pending count into days of customer-visible delay (Section 4.3).

4. The cash-cycle extension beyond the invoice gate, and the exact-decimal requirement every monetary computation in it carries (Sections 4.4 and 6.2).
5. An eight-dimension capability reference framework for manufacturing ERP (Section 6.3).

2 Background and Related Work

Enterprise resource planning has an unusually clear lineage and an unusually contested boundary. The lineage explains why these systems compute what they compute; the boundary explains why they so often fail at the edge of the shop floor.

2.1 From requirements planning to enterprise planning

Orlicky's 1975 statement of material requirements planning is the founding text. Its central move was to derive component demand from a production schedule and a bill of materials rather than forecasting it independently — dependent demand, calculated, not predicted.

Everything since has widened that calculation without changing its shape: manufacturing resource planning added capacity and finance, enterprise resource planning added the rest of the business. The consequence for this paper is that the object at the centre of an ERP is a plan, and a plan is only as good as its ability to be re-derived when something changes.

Harris's economic order quantity sits underneath the same structure, fixing lot size by balancing set-up cost against carrying cost. It is the reason a production sheet quantity is rarely the sales order quantity, and the reason those two numbers appear as separate columns rather than one.

2.2 The boundary the standard actually draws

IEC 62264, adopted from ANSI/ISA-95, separates level 4 business planning from level 3 manufacturing operations and specifies the interface content between them. Part 2 gives the object model that content must satisfy; Part 3 gives the activity models — production, quality, maintenance, inventory — that live at level 3.

The standard is explicit that these are different responsibilities, not different screens of one application. This is exactly the distinction the product's material draws when it says an ERP should be the system of record while high-resolution production data lives where it belongs — a position this paper endorses, because the alternative makes the ERP responsible for data it cannot capture at the required rate.

> A manufacturing ERP that tries to be a machine data system fails at both. The useful question is not how much it captures but whether its interfaces to what does capture are defined.

2.3 Why a pipeline is a queueing system

Little's Law relates the number of items in a system, the rate at which they arrive and the average time each spends there. It holds in steady state for any system with a defined boundary, and an order-processing gate is such a system: orders arrive from the preceding gate, wait, and leave.

This is the mechanism that lets Section 4.3 convert a pending count — a status, seemingly — into a number of days. It is also the reason a pending count that stays constant is not evidence of a healthy gate: constant work in process with a low clearance rate is a long delay, not a stable one.

3 System Overview

The system is organised around one object that moves through five documents. A sales order is raised against a customer purchase order; a production sheet turns it into manufacturing work; despatch, packing and delivery gates move it physically; an invoice closes it commercially. Each document carries the identifier of the one before it, which is what makes the pipeline traversable in both directions.

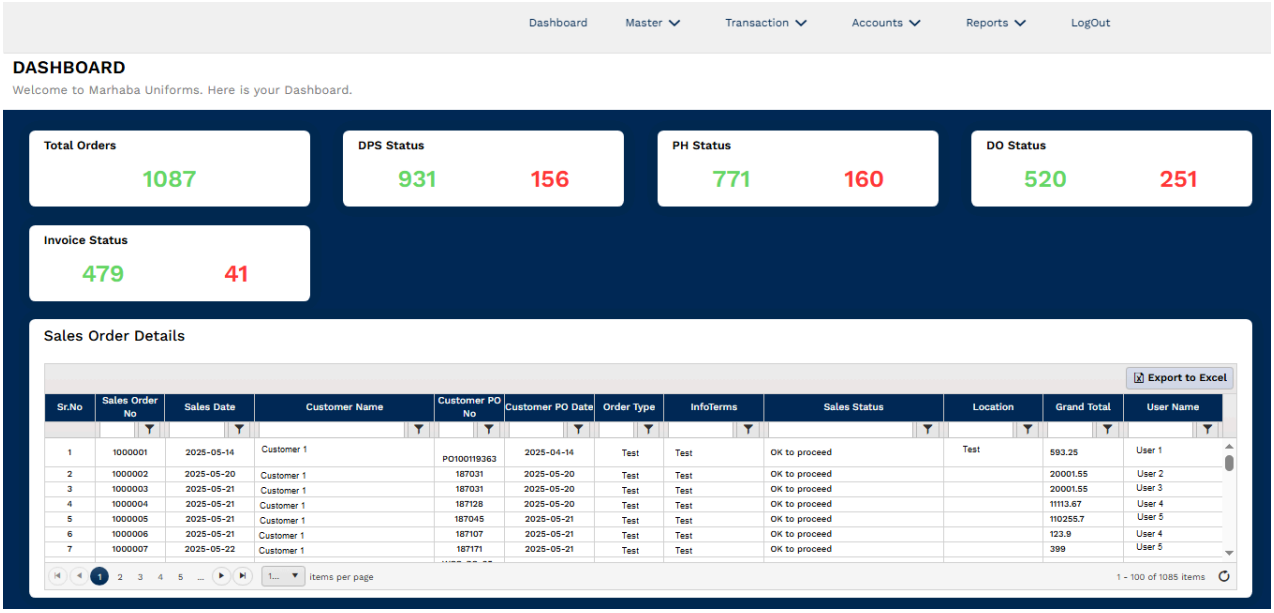

Figure 1. Product capture: the operations dashboard. Total orders sit above four gate tiles, each reporting a cleared count and a pending count, over a sales order grid carrying order number, dates, customer purchase order reference, order type, terms, status, location, value and user. The gate counts are conserved — Section 4.1 states the identity and Section 4.2 reads the constraint out of it.

3.1 The five documents and the four gates

Table 1. The document chain, what each gate asserts, and the link that carries identity forward.

Document	What passing the gate asserts	Link carried
Sales order	The commitment exists, priced, with terms and a promise date	Customer purchase order number and date
Production sheet	The commitment has become manufacturable work, with quantities and dates	Sales order number, on its own production order number
Packing / handover	The work is complete and physically consolidated	Production order reference
Delivery order	The goods have left under the agreed trade terms	Packing reference and destination
Invoice	The commercial claim is raised against the delivery	Delivery order number, retrieved on the invoice

The chain is observable in the captures. A production sheet carries its own order number alongside the sales order number and the customer purchase order number; the invoice screen retrieves its delivery order rather than being typed against a customer. That is the same identity-at-capture discipline that governs traceability, applied to commercial documents.

3.2 Two quantities and two dates on every line

Order lines carry ordered quantity and released quantity as separate columns, and promise date alongside required date. Both pairs exist for the same reason: the number the customer asked for and the number the plant committed to are different facts, and collapsing them destroys the only record of the gap.

The quantity pair is where lot sizing surfaces. Where they differ, the difference is a deliberate planning decision — a minimum batch, a set-up economy, a partial release — and the line records what was decided rather than only what resulted.

The date pair is more consequential, because it is the one a customer feels. A required date is an input; a promise date is a commitment. Section 6.1 argues that the distance between them, aggregated, is a better health measure for a pipeline than on-time delivery against the promise — which measures only whether the plant kept a promise it set itself.

> A plant with perfect on-time delivery against its own promise dates and a systematic two-week gap to required dates is failing its customers with a green dashboard.

3.3 What an order line resolves to

An order line is not always a quantity of an interchangeable item. In the captured deployment — a uniform manufacturer — an invoice line expands to the individual recipients it will be worn by, each with a size and a quantity, so a line for three units resolves to three named people.

This is worth stating because it is the kind of requirement a horizontal ERP handles by adding a free-text field. A per-recipient sub-line is a structural feature: it must survive from order through production sheet to invoice, and it must aggregate correctly to the line total at every gate.

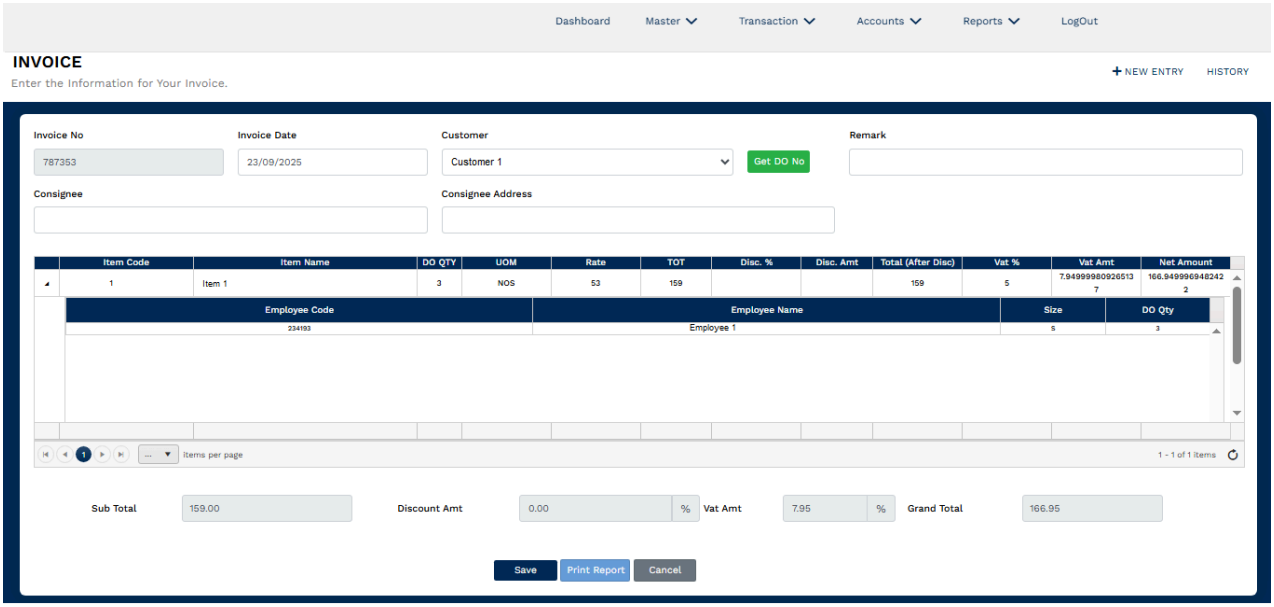

Figure 2. Product capture: the invoice screen. The header retrieves its delivery order rather than being keyed against a customer directly, and the item line expands to per-recipient sub-lines carrying employee code, name, size and quantity — which then aggregate to the line and to the invoice totals shown beneath.

4 Computational Methods

Four computations turn the dashboard of Figure (dashboard) from a status display into an instrument. The first establishes the identity the others rest on.

4.1 The conservation identity

Let the gates be indexed i = 1..k, with n(i) orders cleared at gate i and w(i) orders pending there, and n(0) the count that entered the pipeline.

cleared + pending at each gate = cleared at the gate before it; stage conversion eta(i) = n(i) / n(i-1); end-to-end conversion is the product of the stage conversions (conserve)

n(0) is the total orders raised. The identity holds because an order that has cleared gate i-1 is, by construction, either cleared or pending at gate i — there is no third state and no order enters mid-pipeline.

The identity is verifiable against the captured dashboard. With 1,087 orders raised: 931 + 156 = 1,087 at the first gate, 771 + 160 = 931 at the second, 520 + 251 = 771 at the third, and 479 + 41 = 520 at the fourth. Four independent checks, all exact.

> This is the first thing to test on any pipeline dashboard, and it takes one subtraction per tile. A cascade that does not conserve is counting different populations at different gates, and every conversion figure derived from it is meaningless.

4.2 Locating the constraint

Given conservation, the constraint is the gate accumulating the most work. It requires no cycle-time study, no capacity model and no additional instrumentation — only the counts already displayed.

$$\textit{the constraint gate } i^{*} \textit{ is the one with the largest pending set; stage loss at gate } i \textit{ is the pending share of what arrived there} \qquad \text{(constraint)}$$

the pending set is measured at an instant, so a single observation identifies a candidate rather than proving a persistent constraint. Section 7 treats this limitation explicitly.

Applying this to the captured position gives stage losses of 14.3%, 17.2%, 32.7% and 7.9%. The third gate — delivery — holds 251 orders, more than the other three gates combined hold, and loses nearly a third of what reaches it.

That result is worth pausing on, because it is not where a manufacturer looks. The instinct is that a factory's constraint is production. Here production clears 82.8% of what it receives while delivery clears 67.3%, and any capacity added upstream would simply enlarge the queue at the gate that is actually binding.

Table 2. The captured pipeline position, with stage conversion and loss computed from Equations (conserve) and (constraint). Counts are one deployment at one instant, reported to demonstrate the computation rather than as a benchmark.

Gate	Arrived	Cleared	Pending	Loss
1	1,087	931	156	14.3%
2	931	771	160	17.2%
3	771	520	251	32.7%
4	520	479	41	7.9%
End to end	1,087	479	608	55.9%

End-to-end conversion is the product of the four stage conversions: 0.857 x 0.828 x 0.673 x 0.921 = 0.441. Fewer than half the orders raised have reached an invoice, and the single gate responsible for most of that is the third.

4.3 From a pending count to days of delay

A pending count is not the unit anyone experiences. A customer experiences days, a promise date is set in days, and a plant manager decides in days. Little's Law converts one into the other.

delay at gate i = pending orders at that gate / clearance rate at that gate; total pipeline time is the sum of the queueing delays plus the sum of the service times (flowtime)

lambda(i) is orders cleared per day at gate i, measured over a window in which the rate was stable, and s(i) is the time an order takes to be worked once it is picked up. The decomposition matters because the two terms respond to different interventions.

The separation is the same one that governs response and repair in incident systems: queueing delay is a capacity and sequencing property, service time is a technical one. A gate with a large queue and a short service time needs throughput; a gate with a short queue and a long service time needs the work itself to be faster.

The dashboard supplies the numerator directly. The denominator requires clearance to be counted over a period, which the same document history already holds — so the conversion needs no new data capture, only a query over dates the system stores.

4.4 Where the pipeline actually ends

The dashboard's last gate is the invoice. The business's last gate is the receipt of cash, and between them sits the payment term carried on the sales order header.

order-to-cash time = pipeline time + the payment term; for the observed term, end of month plus ninety days (cash)

the observed term averages 105 days over a uniform distribution of invoice dates within a month. It is one deployment's commercial term, not a product property, but its magnitude relative to the pipeline is the point.

This changes what the dashboard means. If the operational pipeline runs to a few weeks and the payment term adds fifteen, then the gates the system reports are a minority of the cycle that consumes working capital — and an improvement programme aimed only at them is optimising the smaller half.

> None of this argues the operational gates do not matter. It argues that a system displaying four gates and calling the last one the end has drawn its boundary at a convenient place rather than a meaningful one.

4.5 The arithmetic a monetary pipeline requires

Every gate in the pipeline carries money — a line extension, a discount, a tax amount, a total. These are not measurements with tolerances; they are exact quantities in a currency's minor unit, and they must reconcile between a line, a document total and a ledger entry.

$$\textit{each monetary amount is computed and rounded to the currency's minor-unit exponent } e\textit{, and the rounded line amounts must sum exactly to the document total} \qquad \text{(money)}$$

e is the ISO 4217 minor-unit exponent for the currency — 2 where the unit divides into 100, 0 where there is no minor unit, 3 where it divides into 1,000. The identity is what fails when amounts are held in binary floating point.

The requirement is stricter than rounding at display time. A tax amount held as a binary double and rounded only for presentation will differ from the same amount recomputed downstream, and the two will reconcile in most cases and not in some — which is the worst possible failure mode, because it surfaces during an audit rather than during testing.

The rule that follows is simple and worth stating as a requirement for any system of this class: monetary values are stored in exact decimal or in integer minor units, rounded once at the point of computation against the ISO 4217 exponent, and never held in a binary floating-point type at any stage between the order line and the ledger.

5 Reported Outcomes and Field Evidence

Three kinds of claim appear in the product's material, and they are separated here because they carry different weight.

5.1 Observed properties of the software

Table 3. Properties read directly from the delivered software, with where each was observed.

Property	Value	Where observed
Pipeline gates reported	four, with cleared and pending counts each	Dashboard screen
Conservation of gate counts	exact at all four gates	Dashboard screen, verified in Section 4.1
Quantity pair on order lines	ordered quantity and released quantity	Sales order and production sheet screens
Date pair on order lines	promise date and required date	Production sheet screen
Document identity chain	each document retrieves its predecessor's number	Production sheet and invoice screens
Per-recipient sub-lines	employee code, name, size, quantity beneath an item line	Invoice screen
Payment terms on the header	end of month plus 90 days	Sales order screen

None of these is a performance claim. They are structural facts about what the software models, and they are the facts a buyer needs before any benefit statement means anything.

5.2 Figures published for this module

The product's material publishes one timing figure: most implementations complete in twelve to twenty weeks for core production planning, bill of materials, routing and sales-order modules, with integration to external systems adding four to eight weeks depending on complexity.

That is a vendor-reported range with no sample attached, and this paper does not upgrade it. Its usefulness is comparative rather than absolute: the fourteen-month implementation the same material uses as its foil is roughly three times the upper end of the range, and the difference in order of magnitude is the claim being made.

5.3 A deployment in which this module was one of four

A Tier-1 automotive supplier deployed four MileSoft modules together across two plants. This module served as the system of record behind the line — production planning, sales orders, invoicing and production sheets — while the other three captured the high-resolution production data it does not attempt to capture itself.

The programme reported a phased rollout of eight weeks at the first plant and six at the second, the compression attributed to lessons carried across.

> Shared attribution. The rollout figure belongs to the four-module programme, not to this module alone, and the programme's headline outcomes — warranty rework, audit preparation, response time — were produced by the traceability and andon modules rather than by the system of record beneath them.

What can be said specifically is architectural: the deployment is an instance of the boundary Section 2.2 describes, with the ERP holding the commercial and planning record and level 3 systems holding the fastening events, the downtime events and the material genealogy. It is evidence that the boundary is workable, not evidence of what the ERP alone delivers.

5.4 Modelled effort avoided

The published return model for this product is stated as manual data entry and reporting eliminated. Its assumptions are printed here so a reader can substitute their own.

Table 4. Modelled scenario — not a deployment result. Assumptions are the published defaults of the return-on-investment model for this product.

Assumption	Value
People performing manual data entry	10
Hours per person per day	2.5
Working days per year	300
Baseline effort	7,500 hours per year
Share of that effort automated	55%
Modelled hours avoided	4,125 per year

The 55% is an assumption and the parameter to interrogate. What the model represents well is that the effort removed is re-keying between documents — the same order retyped at each gate — which is precisely what the identity chain of Section 3.1 eliminates by construction. What it represents poorly is the offsetting effort a new system creates: master data maintenance, exception handling and the reconciliation that follows any integration.

6 Discussion

6.1 On-time delivery measures the wrong promise

Almost every manufacturer reports on-time delivery against the promise date, and almost every one of them scores well on it. The reason is that the promise date is set by the plant.

The order line carries both dates, so the honest measure is available: the gap between the date the customer required and the date the plant committed to, aggregated across orders. A plant meeting 98% of its own promises while promising two weeks later than asked has a customer problem that its dashboard reports as success.

This is the same structural point as Section 4.4. Both are cases of a boundary drawn where the data is convenient rather than where the question is — and in both cases the system already holds what is needed to draw it correctly.

6.2 Monetary integrity is not a rounding preference

Section 4.5 states the exact-decimal requirement as arithmetic. It is worth saying why it belongs in a paper about pipelines rather than in an implementation note.

A conserved pipeline makes a promise: that the same order is the same object at every gate. Monetary amounts are part of that object. If a tax amount computed at the order line differs in the fifteenth decimal place from the same amount recomputed at the invoice, then the two documents describe different objects,

and the conservation the whole measurement rests on is true of counts but not of values.

The failure is silent by construction. Binary floating point represents most two-decimal values inexactly, so comparisons succeed by luck and fail by luck, and a reconciliation that passes in testing can fail on a production dataset with no code change between them. The fix is architectural rather than defensive: hold monetary values as integer minor units or exact decimals throughout, round once at computation against the ISO 4217 exponent, and never compare monetary values for equality after a floating-point round trip.

> The test is one query: sum the stored line amounts of a hundred invoices and compare each sum to the stored document total, exactly rather than to two places. Any mismatch is a system that will eventually disagree with its own ledger.

6.3 A capability reference framework for manufacturing ERP

Table 5. Capability reference framework. Each dimension separates a manufacturing ERP from a general ledger with a production module, and each is answerable by demonstration on a live pipeline rather than by reading a specification.

Dimension	Question the system must answer by demonstration
D1 Conservation	Do the gate counts on the dashboard satisfy cleared plus pending equals the prior gate's cleared?
D2 Identity chain	Open an invoice. Can you reach the originating customer purchase order without a search?
D3 Two quantities	Where ordered and released quantities differ, does the line record both, and why they differ?
D4 Two dates	Can the system report the gap between required and promise dates across all open orders?
D5 Flow time	Convert a pending count into days of delay. Does the system do it, or is it a spreadsheet exercise?
D6 Change propagation	Release an engineering change. Which revision do in-flight orders show?
D7 Exact money	Sum stored line amounts against the stored document total, exactly. Do they match on every invoice?
D8 The eleven o'clock test	Change the schedule now. When does the floor display show it?

> D1 and D7 take minutes and are almost never asked. Both test whether the system's own numbers agree with each other, which is a lower bar than any feature comparison and a more informative one.

6.4 Generalisability

The conserved-cascade model generalises to any staged process with a fixed sequence and no mid-pipeline entry: claims handling, recruitment, permitting, clinical pathways. The constraint-location result generalises with it, and is at its most useful precisely where intuition points elsewhere.

Two things do not generalise. The captured gate counts are one deployment at one instant, and Section 7 explains why a single observation identifies a candidate constraint rather than proving one. And the payment term is a commercial fact of one customer relationship, not a property of the software.

7 Threats to Validity and Limitations

1. The gate counts are a single snapshot. A pending set observed once identifies a candidate constraint; proving a persistent one requires the same measurement repeated, because a large queue can be the transient result of a batch release rather than a capacity limit.
2. Cumulative counts hide age. The dashboard reports how many orders are pending, not how long each has been pending, so a gate with a few very old orders and one with many recent ones look alike until Equation (flowtime) is applied.
3. Little's Law needs a stable rate. Stage flow times computed across a demand step or a shutdown are not comparable with those either side of it, and the clearance rate must be measured over a window in which it did not shift.
4. The implementation range is vendor-reported. Twelve to twenty weeks carries no sample, no definition of completion and no failure rate, and the fourteen-month figure it is contrasted against is a rhetorical foil rather than a measured comparator.
5. The deployment result is confounded. The eight-week and six-week rollout belongs to a four-module programme, and the outcomes that programme is known for came from the other three modules.
6. The modelled 55% is an assumption. It counts effort removed and not the master-data maintenance, exception handling and reconciliation any such system creates.
7. One deployment's screens are not a product specification. The per-recipient sub-lines, the trade terms and the payment terms are configuration of one installation, and a different customer's screens would show different structures.
8. This paper does not evaluate the modules it does not see. Bill of materials versioning, engineering change propagation and material reservation are described in the product's material and are not visible in any capture, so nothing is claimed about them here beyond what that material states.

> The second limitation is the one that matters most in practice. Age, not count, is what a customer waiting on an order experiences — and a pipeline dashboard that reports only counts is reporting the quantity of the problem rather than its severity.

8 Future Work

- Ageing the pending sets. Reporting the age distribution of orders waiting at each gate, rather than only their count, would turn Section 4.3 from a derivation into a screen and address the limitation this paper considers most serious.
- Constraint tracking over time. Recording the gate with the largest pending set daily would distinguish a persistent constraint from a batch-release artefact, which a single observation cannot.
- Publishing the required-to-promise gap. The data is already on the line; aggregating and reporting it would replace an on-time measure that grades the plant against its own promise.
- Extending the dashboard to cash. Adding a receipt gate after the invoice would show the full order-to-cash cycle rather than the operational majority of it, and would put the payment term where a plant manager can see its cost.
- Isolating this module's contribution in multi-module programmes, so a system-of-record deployment can be evaluated on its own terms rather than through outcomes produced by the modules above it.

9 Conclusion

A manufacturing ERP is judged on features and should be judged on whether its own numbers agree with each other. The order pipeline gives a way to check that in minutes, because it is a conserved cascade: at every gate, cleared plus pending equals what arrived.

Three results follow from the identity. Stage conversion and end-to-end conversion are computable from counts a dashboard already shows. The constraint is the gate with the largest pending set — which, in the position captured here, is delivery rather than production, and capacity added upstream would only enlarge that queue. And a pending count becomes days of customer-visible delay through Little's Law, using dates the system already stores.

Two boundaries in these systems are drawn where the data is convenient rather than where the question is: on-time delivery measured against a promise the plant set itself, and a pipeline declared complete at the invoice when the payment term adds fifteen weeks beyond it. Both can be corrected with data the system already holds, which makes them choices rather than constraints.

The framework of Section 6.3 is offered as the durable contribution. Its first and seventh dimensions cost minutes and ask the same question in two forms: do the system's numbers reconcile to each other, in counts and in money?

Appendix A Nomenclature

Table 6. Symbols and abbreviations used in this paper.

Symbol / term	Meaning
n(0)	Orders raised — the count entering the pipeline
n(i)	Orders cleared at gate i
w(i)	Orders pending at gate i
eta(i)	Stage conversion at gate i — cleared divided by arrived
l(i)	Stage loss at gate i — pending share of what arrived
i*	The constraint gate — the one with the largest pending set
lambda(i)	Clearance rate at gate i, in orders per day
W(i)	Queueing delay at gate i, from Little's Law
s(i)	Service time at gate i — work time once an order is picked up
T(o2c)	Order-to-cash time: pipeline time plus the payment term
tau(terms)	The payment term expressed in days
e	ISO 4217 minor-unit exponent for the currency — 2, 0 or 3
m, M	A rounded line amount and the document total it must sum into
MRP	Material requirements planning, in Orlicky's sense
BOM	Bill of materials
ECO	Engineering change order

Appendix B Worked Numerical Examples

Appendix B.1 Reading the constraint out of five numbers

A pipeline reports 1,087 orders raised and four gates with cleared counts 931, 771, 520 and 479.

Pending counts follow by Equation (conserve): 1,087 — 931 = 156, 931 — 771 = 160, 771 — 520 = 251, 520 — 479 = 41. Stage conversions are 0.857, 0.828, 0.673 and 0.921; stage losses 14.3%, 17.2%, 32.7% and 7.9%.

By Equation (constraint) the constraint is the third gate, holding 251 orders — more than the 156, 160 and 41 at the other three combined. End-to-end conversion is 0.857 x 0.828 x 0.673 x 0.921 = 0.441.

Adding 20% capacity at the second gate would move 34 more orders per period into a queue already 251 deep. The whole of that investment would appear as a larger pending count at the third gate, and end-to-end conversion would not move.

Appendix B.2 The same pipeline in days

Suppose the four gates clear 62, 51, 26 and 48 orders per working day, measured over a stable month, with service times of 0.5, 2.0, 0.5 and 0.2 days.

Applying Equation (flowtime): W1 = 156 / 62 = 2.5 days, W2 = 160 / 51 = 3.1, W3 = 251 / 26 = 9.7, W4 = 41 / 48 = 0.9. Queueing delay totals 16.2 days; service time totals 3.2; total pipeline time is 19.4 days.

The decomposition changes the diagnosis. Gate 2 has the longest service time at 2.0 days — it is the real manufacturing work — but contributes only 3.1 days of waiting. Gate 3 does almost nothing to an order and makes it wait 9.7 days, which is half the pipeline. One of these is a technical problem and the other is a sequencing and capacity problem, and only the decomposition tells them apart.

By Equation (cash), with an average payment term of 105 days, order-to-cash is 124.4 days. The nineteen operational days the dashboard reports are 16% of it.

Appendix B.3 Why the arithmetic has to be exact

An invoice line carries quantity 3 at 53.00 per unit, giving 159.00, with tax at 5%. In a currency with a minor-unit exponent of 2, Equation (money) gives a tax amount of 7.95 and a net of 166.95.

Computed in binary double precision, 159.00 x 0.05 is not 7.95. The nearest representable double is 7.949999999999999289457264239899 8, and the net is 166.949999999999988631316227838 40. Rounded for display both look correct.

The failure appears at reconciliation. Sum a hundred such lines and the accumulated error can exceed half a minor unit, so the stored document total and the sum of the stored line amounts differ by a cent — and the two disagree exactly when a ledger, an auditor or a tax authority compares them. Computing in integer minor units — 15,900 x 5 / 100 = 795 — makes the identity exact by construction, and no rounding policy is required at any later stage.

The error is not large. It is unpredictable, which is worse: a system that is wrong by a cent on one invoice in three hundred cannot be trusted on any of them without checking, which defeats the point of the system.

Data provenance

Every quantitative claim in this paper resolves to one of the entries below. Figures marked as modelled estimates are not deployment measurements; their assumptions are stated.

Metric	Reported value	Provenance
Order pipeline position across the four gates on the dashboard	1,087 orders; 931 / 771 / 520 / 479 cleared, 156 / 160 / 251 / 41 pending	**Product specification** /products/erp — Read from the dashboard screen captured on the product page; each gate's cleared plus pending equals the previous gate's cleared count.
Implementation time for core modules	12-20 weeks	**Product specification** /products/erp
Payment terms carried on the sales order header	EOM90DAYS — end of month plus 90 days	**Product specification** /products/erp — Read from the sales order screen captured on the product page; one deployment's terms, not a product default.
Hours of manual data entry automated per year	4,125 h (modelled)	**Modelled estimate** 10 people spending 2.5 hours per day on data entry, across 300 working days; 55% of that effort automated; Model and defaults published in src/data/roiModels.ts
Phased rollout duration	8 weeks (plant 1), 6 weeks (plant 2)	**Operator-reported — Tier-1 Manufacturers (Pune + Nasik plants, India)** /case-studies/industry40

References

[1] J. Orlicky (1975). Material Requirements Planning: The New Way of Life in Production and Inventory Management. McGraw-Hill, New York; the founding text of MRP and of every ERP descended from it.

[2] F. W. Harris (1913). How Many Parts to Make at Once. Factory, The Magazine of Management, vol. 10, no. 2, pp. 135-136, 152; the first statement of the economic order quantity, reprinted in Operations Research 38(6), 1990, pp. 947-950. https://pubsonline.informs.org/doi/10.1287/opre.38.6.947

[3] Little, J. D. C. (1961). A proof for the queuing formula: L = lambda W. Operations Research, 9(3), 383-387.

[4] International Electrotechnical Commission / International Society of Automation (2025). Enterprise-control system integration - Part 1: Models and terminology. IEC 62264-1; ANSI/ISA-95.00.01-2025 (IEC 62264-1 Mod).

[5] International Electrotechnical Commission (2015). IEC 62264-2 - Enterprise-control system integration - Part 2: Objects and attributes for enterprise-control system integration. IEC, Geneva; specifies the generic interface content exchanged across the boundary between level 3 manufacturing systems and level 4 business systems. https://www.iso.org/standard/57310.html

[6] International Society of Automation (2013). Enterprise-control system integration - Part 3: Activity models of manufacturing operations management. ANSI/ISA-95.00.03-2013 (IEC 62264-3 Modified).

[7] International Electrotechnical Commission (2025). IEC 62541-1: OPC unified architecture - Part 1: Overview and concepts. IEC, Geneva. https://webstore.iec.ch/en/publication/81513

[8] International Organization for Standardization (2014). ISO 22400-1: Automation systems and integration - Key performance indicators (KPIs) for manufacturing operations management - Part 1: Overview, concepts and terminology. ISO, Geneva. https://www.iso.org/standard/56847.html

[9] International Organization for Standardization (2014). ISO 22400-2: Automation systems and integration - Key performance indicators (KPIs) for manufacturing operations management - Part 2: Definitions and descriptions. ISO, Geneva; defines OEE as availability x effectiveness x quality ratio. https://www.iso.org/standard/54497.html

[10] International Organization for Standardization (2015). ISO 4217:2015 - Codes for the representation of currencies. ISO, Geneva; defines the three-letter alphabetic and three-digit numeric currency codes and, for currencies with minor units, the decimal relationship between the minor unit and the currency. https://www.iso.org/standard/64758.html

[11] International Chamber of Commerce (2019). Incoterms 2020: ICC Rules for the Use of Domestic and International Trade Terms. ICC Publication No. 723E, Paris; launched September 2019 and in force from 1 January 2020, defining the eleven trade terms that allocate cost and risk between seller and buyer.

[12] International Automotive Task Force (2016). Quality management system requirements for automotive production and relevant service parts organizations. IATF 16949:2016, first edition, 1 October 2016 (superseding ISO/TS 16949).

[13] GS1 (2024). GS1 General Specifications. GS1 AISBL; defines the GTIN, SSCC and GLN identification keys. https://ref.gs1.org/standards/genspecs/

[14] GS1 (2024). Serial Shipping Container Code (SSCC). GS1 identification key series; an 18-digit key for a logistic unit, carried under application identifier 00. https://www.gs1.org/standards/id-keys/sscc

[15] International Organization for Standardization / International Electrotechnical Commission (2014). ISO/IEC 15459-1: Information technology - Automatic identification and data capture techniques - Unique identification - Part 1: Individual transport units. ISO/IEC, Geneva. https://www.iso.org/standard/54779.html

[16] International Organization for Standardization / International Electrotechnical Commission (2024). ISO/IEC 19987: Information technology - EPC Information Services (EPCIS) Standard, version 2.0. ISO/IEC, Geneva. https://www.iso.org/standard/85557.html

[17] H. F. Dickie (1951). ABC Inventory Analysis Shoots for Dollars, Not Pennies. Factory Management and Maintenance, vol. 109, no. 7, July 1951, pp. 92-94; the origin of ABC classification, applying Pareto and Lorenz to stock value.

[18] Nakajima, S. (1988). Introduction to TPM: Total Productive Maintenance. Productivity Press, Cambridge, MA.

[19] Kagermann, H., Wahlster, W., and Helbig, J. (2013). Securing the future of German manufacturing industry: Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final report of the Industrie 4.0 Working Group, acatech - National Academy of Science and Engineering. https://www.acatech.de/wp-content/uploads/2018/03/Final_report__Industrie_4.0_accessible.pdf

[20] Lee, J., Bagheri, B., and Kao, H.-A. (2015). A cyber-physical systems architecture for Industry 4.0-based manufacturing systems. Manufacturing Letters, 3, 18-23.