SIMCEL



ONE MODEL, ONE TRUTH

OPTIMAL, AND STILL WRONG

The Shift from Static Optimisation to Dynamic Simulation and why the difference determines what your planning system can and cannot do

Julien Brun
Founder & CEO

www.simcel.io
info@simcel.io

Table of contents

EXECUTIVE SUMMARY: TWO ARCHITECTURES. ONE MISUNDERSTOOD MARKET.

1. INTRODUCTION: SIMULATION MODELS BEHAVIOUR. OPTIMISATION FINDS ALLOCATIONS.

- 1.1 The central distinction
- 1.2 Why this paper exists
- 1.3 A note on neutral ground
- 1.4 What architecture actually determines

2. THE DATA LAYER: WHAT A KNOWLEDGE GRAPH ACTUALLY DOES

- 2.1 Graph database fundamentals
- 2.2 Knowledge graphs in planning
- 2.3 What graphs enable
- 2.4 What graphs don't provide: the map, not the journey
- 2.5 The o9 example: a modern data layer on a familiar decision layer

3. OPTIMISATION-BASED PLANNING: HOW IT WORKS AND WHERE IT EXCELS

- 3.1 The mathematical engine
- 3.2 The aggregation requirement
- 3.3 Cost coefficients and the linear assumption
- 3.4 When optimisation-based planning is the right choice
- 3.5 Scenarios vs simulation: the critical distinction

4. MULTI-PARADIGM SIMULATION: A DIFFERENT COMPUTATIONAL APPROACH

- 4.1 Three paradigms, one model
- 4.2 System Dynamics: policy-driven flows
- 4.3 Agent-Based Modelling: distributed decisions
- 4.4 Discrete Events: from Transactions to Financials
- 4.5 The Atomizer Algorithm: the strategic-to-tactical bridge

Table of contents

5. THE OPTIMISATION QUESTION: IF SIMULATION DOESN'T OPTIMISE, HOW DO WE KNOW WE'RE MAKING THE RIGHT CHOICES?

- 5.1 When single-objective optimisation makes sense
- 5.2 Why strategic decisions resist single-objective optimisation
- 5.3 Simulation-optimisation: see the trade-off, then choose

6. THE ARCHITECTURAL CHOICE: WHICH APPROACH FITS WHICH PROBLEM

- 6.1 Two kinds of artifact
- 6.2 When graph-based optimisation is the right choice
- 6.3 When multi-paradigm simulation is the right choice
- 6.4 Integration where the network is truly global

CONCLUSION: THE QUESTION THIS PAPER ANSWERS

THE OPENING QUESTION

ABOUT THIS SERIES: ONE MODEL, ONE TRUTH

EXECUTIVE SUMMARY

Two *architectures*. One misunderstood market.

The enterprise planning technology market rests on a widespread and consequential misconception: that optimisation-based planning platforms and multi-paradigm simulation are competing alternatives for the same job. They are not. They are two different computational approaches to the same business problem, each genuinely suited to different conditions.

The confusion is sharpened by how the market describes itself. The most visible platforms compete on their data layer, the way they store and connect the network, and the knowledge graph is the current headline. That is a real advance in how a planning model is represented. It is not an advance in how the plan is decided. Beneath every one of these platforms, the plan is still produced the same way: by solving for an allocation on an aggregated, coefficient-based model. That decision layer, not the data layer the marketing competes on, is where the distinction that matters lives.

The cost runs in both directions. Organisations deploying optimisation for problems that require behavioural modelling find their plans break down between strategy and execution. Organisations hesitating to deploy simulation because they believe optimisation is always faster or more rigorous leave precision and decision confidence on the table.

The distinction itself is not new. Operations research has separated optimising an allocation from simulating behaviour for decades. What has not been done is to carry that distinction into the decision corporate buyers actually make: which architecture to build planning infrastructure on, and what that choice silently determines about financial precision, horizon fidelity, and the ceiling for AI-enabled autonomy.

Beneath all of this is a problem most planning organisations live with: the plan and the financials never fully reconcile, and the gap is closed by hand, every cycle. This work argues that gap is architectural, not a matter of process. This paper examines how each architecture actually works, where each genuinely excels and where each fails, and what determines when each is the right choice. It also looks closely at the knowledge graph, the category's most-marketed innovation and the foundation of its most modern platform, to separate what it genuinely does from what its positioning implies.

The central finding is direct: these are not competitors.

Optimisation-based planning

dominates large-scale volume allocation with single objectives and stable cost structures.

Multi-paradigm simulation

dominates wherever behaviour, granularity, financial precision, or multi-objective trade-offs matter.

Understanding the boundary between them is the most practically useful insight a technology decision-maker can have before committing to planning infrastructure.

This paper examines:

- What a knowledge graph genuinely does, using o9 as the leading example, and why it is a data-layer advance rather than a decision-layer one.
- How optimisation-based planning actually works, and the aggregation and linear-cost assumptions it requires.
- The three-paradigm architecture of multi-paradigm simulation: System Dynamics, Agent-Based Modelling, and Discrete Event Simulation, and what each contributes.
- The critical distinction between scenario re-optimisation and genuine simulation.
- A worked example showing what simulation-optimisation finds that pure optimisation misses.
- A selection framework for which architecture fits which planning problem

INTRODUCTION

1. Simulation models behaviour. Optimisation finds allocations.

1.1 The Central Distinction

The single most important sentence in enterprise planning architecture is also the most frequently overlooked: simulation models behaviour. Optimisation finds allocations.

This is not a subtle technical distinction. It describes a fundamental difference in what these two computational approaches actually compute and why the difference has direct consequences for the business decisions organisations make with them.

OPTIMISATION-BASED PLANNING

Finds the mathematically optimal allocation of resources at a point in time, subject to defined constraints.

> Given what we know right now, what is the best way to distribute available capacity across demand?

MULTI-PARADIGM SIMULATION

Models how the business behaves as a set of decisions unfold over time with variability, feedback loops, sequential events, and non-linear cost dynamics.

> Given how our supply chain actually operates, what will actually happen if we make this choice?

Both questions matter. Neither answer is a substitute for the other.

1.2 Why this paper exists

Most planning organisations live with a quiet, recurring problem: the plan that comes out of the supply-chain and S&OP system and the numbers that come out of the FP&A and finance system never quite agree, and the gap is reconciled by hand, in spreadsheets, every cycle. The first paper in this series argued that this gap is architectural rather than procedural. It cannot be closed by better process layered on top of two disconnected data models, because the two models were never built to produce the same truth.

That diagnosis sets up the question this paper answers. The architecture a business plans on controls five things that determine what any plan can actually deliver: how granular the plan can be, how precise its financials are, whether it holds fidelity as it moves between time horizons, how far demand, supply, and finance share a single model, and whether it can support autonomous, AI-driven planning. These five run through everything that follows, and Section 1.4 sets them out in full.

This paper examines both architectures in depth, gives an honest account of where each genuinely excels and where each fails, and ends with a framework for deciding which approach fits which planning problem.

1.3 A note on neutral ground

The optimisation-based planning platforms in wide use today, o9 Solutions, Kinaxis, Blue Yonder, and SAP Integrated Business Planning, have delivered real value over two decades and are mature, proven technology. They differ in how they store and connect the planning model: o9 is built on a knowledge graph, Kinaxis on a concurrent in-memory model, SAP IBP on an in-memory column store. They converge on how they decide. Each produces the plan by solving for an allocation, through mathematical optimisation, heuristics, or both. It is that shared decision layer, not the data layer their marketing competes on, that this paper sets against simulation.

The distinction itself is old. Operations research has separated optimising an allocation from simulating behaviour for decades, and the trade-off between the two is well studied. What has stayed inside the academic literature is the consequence for the people who buy planning systems. This paper does not claim to discover the distinction. It carries it to an architectural decision that has, until now, been made on feature comparisons rather than on what each approach can structurally do.

The goal is not to argue these platforms should be replaced universally. The goal is clarity: when is each approach the right architectural choice, and why?

1.4 What architecture actually determines

Before examining either architecture, it is worth being precise about what the architectural choice controls, because this is what makes the distinction matter beyond technical preference.

Every planning platform is two layers. A data layer, how it stores and connects the network: products, plants, customers, lanes, capacities. And a decision layer, how it produces the plan: by solving for an allocation, or by simulating behaviour over time. The two are independent. A knowledge graph, a concurrent in-memory model, and a column store are all choices at the data layer. Optimisation, heuristics, and simulation are choices at the decision layer. Most of the market's noise is about the data layer. Almost everything that determines what a plan can tell you is set at the decision layer.

The decision layer determines:

Planning granularity

Whether the system plans at the level of product families and customer segments, or at the level of individual SKUs and specific customers.

Financial precision

Whether costs and margins are approximated through fixed per-unit coefficients, or calculated from actual transaction-level dynamics including overhead absorption, utilisation effects, and batch economics.

Time horizon fidelity

Whether the system degrades in precision when moving between planning horizons, or operates at the same granularity from strategic through to operational.

AI potential

Whether the architecture can support autonomous planning agents operating continuously, or is structurally constrained to periodic re-optimisation.

Integration scope

Whether demand, supply, and finance share a single data model, or require separate tools with manual reconciliation.

Each of these is set by the decision layer at the system's foundation. Not by configuration, not by feature sets, and not by the quality of the implementation. There is one further property the decision layer settles, and the paper returns to it at the end: whether one approach can contain the other. It can. An optimisation can run inside a simulation; a simulation cannot run inside an optimiser. That asymmetry is what finally orders the two.

THE DATA LAYER

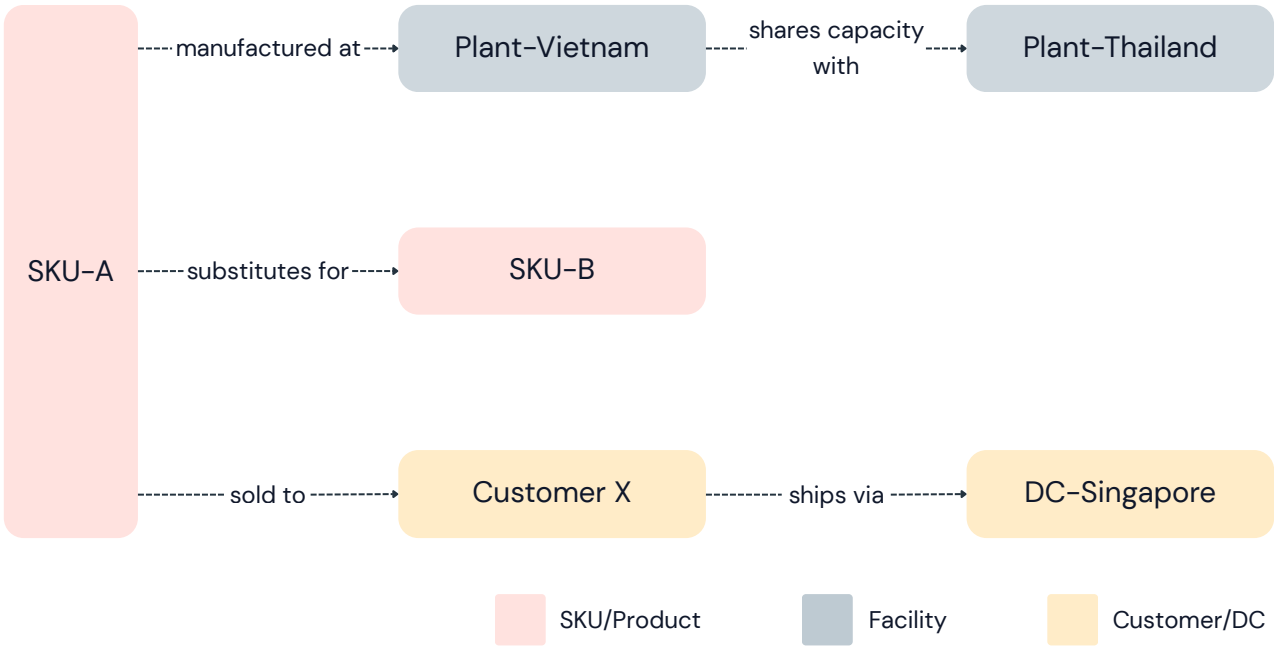
2. What a knowledge graph actually does

2.1 Graph Database Fundamentals

Before examining graph-based optimisation, it is necessary to understand the foundational technology it builds upon: graph databases and knowledge graphs.

A graph database stores data as entities (nodes) and relationships (edges), contrasting with traditional relational databases that organize data in tables. The key differentiator is that relationships are first-class citizens with semantic meaning. Each edge carries explicit business significance, not just a foreign key reference.

For supply chain planning, a graph structure looks like this:



2.2 Knowledge graphs in planning

A knowledge graph extends this structure with a defined ontology. A data model describing the types of entities and relationships relevant to the domain.

For supply chain planning, this typically includes:

Domain Ontology Schema

Entity types (nodes)

- Products, SKUs, product families
- Manufacturing facilities, distribution centres
- Customers, suppliers
- Transportation routes, resources

Relationship types (edges)

- Manufacturing relationships (produced at, requires component)
- Distribution flows (ships from, delivers to)
- Substitution patterns (substitutes_for, can_replace)
- Capacity sharing (shares line with, competes for capacity)

Planning vendors use knowledge graphs to enable network dependency discovery, automatically identifying multi-hop supply paths, detecting hidden dependencies through shared capacity pools, and tracing the downstream impact of a supplier disruption. O9 Solutions, the most prominent example, markets its Enterprise Knowledge Graph as the core of its platform.

EXAMPLE QUERY

Find all products dependent on Vietnam manufacturing that serve customers requiring 98%+ service levels" executes in milliseconds by traversing the graph structure.

2.3 What graphs enable

Knowledge graphs excel at four things:

Relationship discovery

Finding hidden dependencies across complex networks that would require expensive recursive queries in relational databases.

Structural queries

Answering questions about which entities are connected through specific relationship paths.

Impact visualization

Highlighting which nodes are affected when one entity changes - a supplier disrupted, a plant closed, a distribution centre added.

Semantic clarity

Making business logic explicit in relationship definitions rather than embedded in code.

2.4 What graphs don't provide: the map, not the journey

Here is the boundary that matters most:

> Knowledge graphs show structure. They do not model behaviour.

A knowledge graph maps "Product X is manufactured at Plant Y."
It does not calculate:

> *"What happens to unit margin if Plant Y runs at 70% utilisation for a maintenance month?"*

It identifies that Customer A depends on DC-Singapore. **It does not quantify the cash flow impact of a two-week delay in that shipment.**

Knowledge graphs are, in the most precise sense, the map, not the journey. They capture the topology of the supply chain with clarity and speed. What they cannot do is model what happens when decisions are made and time passes.

They identify dependencies but don't quantify financial impacts. They show network structure but not network behaviour. They are the map, not the journey.

This distinction becomes critical when examining the optimisation systems that build upon this foundation - because the limitation of the graph carries through into the optimisation layer in ways that directly affect the business decisions planners can make with confidence.

2.5 The o9 example: a modern data layer on a familiar decision layer

It is worth grounding this in the platform most often held up as the modern standard. o9 Solutions is the newest of the major planning platforms and is widely perceived as the most innovative, largely on the strength of its Enterprise Knowledge Graph, presented as a "Digital Brain" that connects demand, supply, commercial, and financial data into one model.

Strip the framing and the knowledge graph is exactly what the previous pages described, and a good version of it. Holding the whole planning model in one connected semantic structure is a genuine advance over the siloed schemas it replaces, and it makes dependency discovery, tracing what a disruption touches across the network, fast and native. That value is real.

What the graph does not do is decide. When o9 produces a plan, it does not reason over the graph to reach an answer. It aggregates data out of the graph and hands it to a mathematical optimiser and a set of heuristics working from cost coefficients, the decision layer the next section examines. The graph feeds the solver. It does not replace it, and it does not change what the solver can see.

So o9's modernity is real, and it is located in the data layer. Its decision layer is the same allocation on an aggregated, coefficient-based model that its older peers use, and it inherits the same boundary. The map-not-the-journey limit applies to the most modern platform in the category as fully as to any other.

> **A better map does not make the journey.**

The point is not that the knowledge graph is marketing. It is that a data-layer advance is routinely presented as though it changed what the plan can compute, and it does not. Recognising which layer an innovation lives in is the first defence a buyer has against paying for the wrong one.

OPTIMISATION-BASED PLANNING

3. How it works and where it excels

3.1 The Mathematical Engine

Optimisation-based planning platforms decide by solving. At the core of each sits a solver, typically Linear Programming or Mixed-Integer Programming, or a set of heuristics that approximate one, which finds the allocation that best satisfies a defined objective subject to constraints. This is the decision layer that o9, Kinaxis, Blue Yonder, and SAP Integrated Business Planning share, whatever sits beneath them at the data layer.

The basic structure:

Objective Function (minimise):

Total_Cost = Σ (Production_Cost + Transport_Cost + Inventory_Cost)

The solver explores millions of combinations to find the allocation that minimises total cost or maximises margin, service level, or another defined objective, while respecting all constraints.

When planners create scenarios, the system re-runs optimisation with modified parameters.

SCENARIO EXAMPLE

Consider a baseline where Vietnam runs at 100,000 units/month and Thailand at 130,000 units/month against a 95% service target. The solver produces an allocation: 100,000 units from Vietnam, 85,000 from Thailand, total cost $2.15M, service level 95.2%.

Reduce Vietnam capacity to 80,000 for a maintenance month. Re-optimise. The solver shifts volume: 80,000 from Vietnam, 105,000 from Thailand, total cost $2.28M, service 95.1%.

Each scenario is an independent optimisation run with different coefficients. There is no connection between scenarios. No learning, no progression through time.

3.2 The aggregation requirement

To make optimisation computationally tractable, data must be aggregated. This is not a configuration choice. It is a mathematical necessity.

A problem with 200 products × 10 facilities × 12 months already contains 24,000 decision variables. Maintaining SKU-level or customer-level detail in a global solver would make the problem intractable.

The aggregation cascade is consistent across all three planning dimensions:

PRODUCT

Individual SKUs (2,000 SKUs) → **Product families (50 product families)**

Lost: SKU-specific cost structures, customer-specific product preferences, batch size economics.

TIME

Daily or weekly (365 days) → **Monthly buckets (12 periods)**

Lost: intra-month demand dynamics, actual order timing, shelf-life constraints.

CUSTOMER

Individual accounts (500 customers) → **Segments or regions (20 segments)**

Lost: customer-specific service agreements, payment terms, relationship value.

This aggregation is mathematically necessary but creates a fidelity gap when comparing the plan to operational reality.

3.3 Cost coefficients and the linear assumption

Optimisation uses cost coefficients, fixed per-unit costs for each activity. Vietnam production costs $10.00 per unit. Thailand production costs $11.50 per unit. Transport from Vietnam to Singapore costs $0.80 per unit.

The total cost calculation is linear:

Total_Cost = (Vietnam_Qty × $10.00) + (Thailand_Qty × $11.50) + transport costs.

The critical assumption is linearity. Whether you produce 50,000 units or 100,000 units at the Vietnam plant, the per-unit cost remains $10.00. This assumption holds when costs are genuinely linear (commodities, utilities, straightforward variable costs). It breaks down when:

- Overhead absorption varies with volume (fixed costs spread over fewer units = higher true unit cost)
- Utilisation affects efficiency - overtime, setup costs, ramp ups
- Batch size economics apply - minimum order quantities, container fill rates
- Production mix changes affect shared resource costs

That $10.00 is not a neutral fact about the plant; it is a standard cost, and it is only true at one volume. It is built by spreading the plant's fixed overhead across its planned output: $400,000 of monthly overhead over 100,000 planned units is $4.00 of overhead per unit, on top of $6.00 of variable cost. Run the plant below that planned volume and the same overhead spreads over fewer units, so the true per-unit cost rises while the coefficient does not. At 80,000 units the overhead becomes $5.00 per unit and the true cost $11.00, but the optimiser still books $10.00. The $80,000 gap below is that difference, $1.00 across 80,000 units, overhead the plan quietly failed to absorb.

OPTIMISER VIEW (STANDARD RATE)

ASSUMED UNIT COST

$10.00 / unit

Linear assumption: per-unit cost remains the same regardless of volume.

ACTUAL REALITY (ABSORPTION)

TRUE UNIT COST @ 80K UNITS

$11.00 / unit

$5/unit overhead ($400k ÷ 80k)
+ $6/unit variable cost.

WHAT THIS MEANS IN PRACTICE - 80,000 UNITS, $400,000/MONTH OVERHEAD, $6/UNIT VARIABLE

THE RESULTING GAP

$80,000 Hidden Cost

Invisible to the optimiser for a single production run.

3.4 When optimisation-based planning is the right choice

This architecture is genuinely powerful, and the honesty this paper promised requires stating clearly when it is the right approach:

Large-scale tactical allocation

Across networks with many nodes and clear single objectives

Stable cost structures

Where linear assumptions hold - commodities, fixed logistics rates, predictable labor

Monthly planning cycles

Where aggregation to product families and customer segments is acceptable

Well-defined variable trade-offs

Decisions where the trade-off is between a small number of well-defined variables, minimise cost subject to service constraint

Strategic network design problems

Plant locations, distribution centre footprints - where detailed dynamics are less critical than network structure

Leading vendors have refined this approach over 20+ years. The platforms are mature, the solver technology is sophisticated, ERP integrations are extensive, and scalability to global networks is proven. The question is not whether this architecture is valid, it clearly is, with hundreds of successful implementations. The question is what its boundaries are.

3.5 Scenarios vs simulation: the critical distinction

A common misconception, one that costs organisations significant decision quality, is that scenario capabilities in optimisation-based tools constitute simulation. They do not.

When a planner creates a "scenario" in a graph-based system, the solver re-runs its optimisation with modified parameters, producing a new static optimal allocation.

This is re-optimisation, not simulation.

What re-optimisation cannot do is model how the supply chain behaves as that scenario unfolds over time:

- How inventory depletes week by week as orders are fulfilled and replenishment lags
- How backlogs accumulate when demand spikes exceed available capacity
- How lead time variability compounds across multiple tiers of the network
- How a margin impact materializes differently depending on the sequence of events

A re-optimised scenario shows the best allocation for a set of parameters at a point in time. A simulation shows what actually happens when a decision is implemented and time passes - including second-order effects, timing dependencies, and operational constraints that the linear model cannot capture.

Simulation models behaviour. Optimisation finds allocations.

Real business decisions involve dynamics like timing, variability, interdependence, that a static optimal snapshot systematically obscures.

The question:

THE QUESTION

"What happens if we shift 30% of our Vietnam volume to Thailand for Q3?"

RE-OPTIMISATION ANSWER

Lower cost by $X, service maintained at Y%

SIMULATION ANSWER

Inventory depletes by Week 6 for SKU-A, a $45k expedite cost materializes in Week 8, the margin impact over 13 weeks is Z%

These are not the same answer.

MULTI-PARADIGM SIMULATION

4. A different computational approach

4.1 Three paradigms, one model

Multi-paradigm simulation is a fundamentally different decision layer. Rather than solving for an allocation, it models the full dynamic behaviour of the business, demand, supply, and finance, within a single unified model.

Combining the three established simulation paradigms, System Dynamics, Agent-Based Modelling, and Discrete Event Simulation, in one model is not itself new; general-purpose simulation tools have done it for two decades. What is new is making that combination a planning system rather than a modelling environment. SIMCEL is built natively for integrated business planning. The three paradigms run on one data model purpose-built for demand, supply, and finance, and the financial P&L emerges from the same simulation rather than from a separate tool placed alongside it. The technical definition is unchanged: SIMCEL is a Discretized System Dynamics model with Agent-Based extensions and Discrete Event execution. What separates it from a general-purpose toolkit is what it is built to do.

Each paragraph below is retained: "Each paradigm contributes a distinct capability", then the System Dynamics, Agent-Based Modelling, and Discrete Event Simulation descriptions, then "All three layers operate on a single unified data model."

SYSTEM DYNAMICS (SD)	provides the structural framework, stocks, flows, feedback loops. This is the layer that models how the entire network breathes: inventory building and depleting, demand signals propagating upstream, replenishment orders triggering production.
AGENT-BASED MODELLING (ABM)	provides behavioural heterogeneity. Each facility is an autonomous agent with its own state, decision logic, and interaction protocols. DC-1 might run weekly periodic review with high safety stock. DC-2 runs daily continuous review with lean inventory. DC-3 replays historical patterns. This diversity models real operational variety rather than imposing uniform policies across the network.

DISCRETE EVENT SIMULATION (DES) provides operational precision. Individual transactions (customer orders, shipments, production runs) are processed as events at discrete time points. Each transition involves explicit resource checking, queue management, and constraint enforcement.

All three layers operate on a single unified data model.

4.2 System Dynamics: policy-driven flows

The SD layer models material flows and feedback loops. Stock variables track inventory in five states: on-hand (available to sell), in-transit (moving between locations), on-order (requested from suppliers), conditional hold (quality testing, maturation), and allocated (committed to specific orders).

Flow dynamics follow:

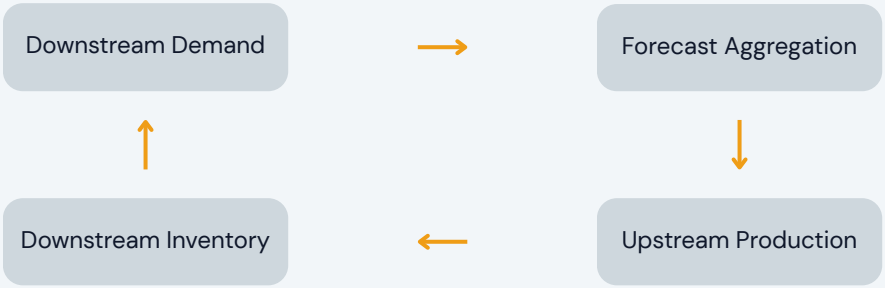
Inventory(t+1) = Inventory(t) + Inflows - Outflows.

The feedback loops this enables are what graph optimisation cannot model.

THE PRIMARY INVENTORY CONTROL LOOP:



THE SECONDARY FORECAST FLOW LOOP:



These feedback loops create systemic patterns - bullwhip effect, inventory oscillations from review period delays - that are invisible in a static optimal allocation but determine whether a plan is executable in practice.

4.3 Agent-Based Modelling: distributed decisions

Each facility in a simulation model is an autonomous agent with state variables (inventory levels, capacity, current policies), decision logic (how it makes production planning, replenishment, and order fulfillment decisions), and interaction protocols (how it communicates with other agents through orders and shipments).

The behavioural heterogeneity this enables is critical for realistic modelling. Not all distribution centres follow identical policies. Different lead time profiles, different demand patterns, different shelf-life constraints, different customer service agreements - all of these can be modelled as distinct agent behaviours within the same simulation run.

EXAMPLE QUESTION

This is what allows the simulation to answer questions that optimisation cannot: "If we apply a continuous review policy at DC-Singapore but maintain periodic review at DC-Jakarta, what is the impact on system-wide inventory levels and service performance over 13 weeks?"

4.4 Discrete Events: from Transactions to Financials

This is the dimension where the distinction from optimisation-based planning has the most direct financial consequence.

Optimisation uses cost coefficients. The Vietnam plant produces at $10/unit - whether it runs at 50,000 units or 100,000 units. Simulation calculates from actual transactions.

EXAMPLE

Consider the Vietnam factory with $400,000/month fixed overhead and $6/unit variable cost:

At 100,000 units:

Overhead/unit = $400k ÷ 100k = $4.00
Total unit cost = $10.00
No variance

At 70,000 units:

Overhead/unit = $400k ÷ 70k = $5.71
Total unit cost = $11.71
$120k under-absorption

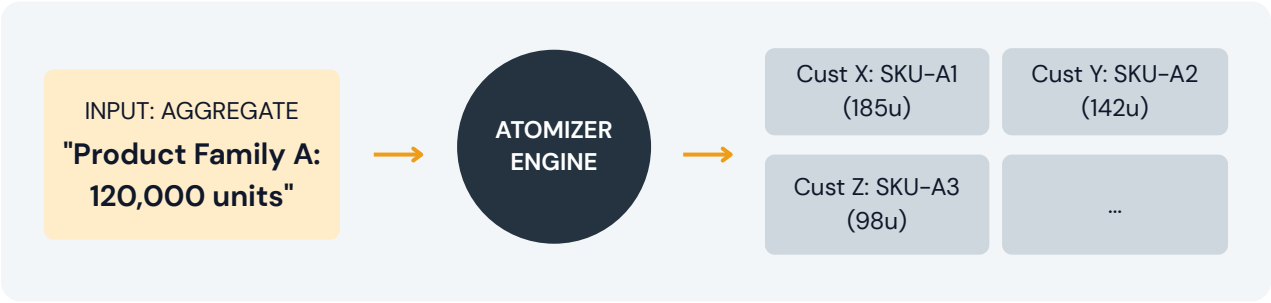
The same facility has materially different unit economics depending on utilisation. A plan that calls for 70,000 units because it was optimal for the network-level allocation produces a cost outcome that the optimiser could not see. The optimiser saw $10/unit. The business delivered $11.71/unit.

Extend this across revenue recognition, working-capital timing, and FX, and the gap between what optimisation reports and what finance actually records stops being a rounding error. It compounds, one dimension at a time, into a material distortion of reported margin: the 32% variance that Paper 1 introduces and Paper 4 takes apart.

4.5 The Atomizer Algorithm: the strategic-to-tactical bridge

A unique capability of SIMCEL's architecture addresses a practical challenge: strategic planners think in aggregates ("5% market growth", "3% cost inflation") while simulation requires transaction-level detail (customer-product-date demand).

The Atomizer solves this by decomposing aggregate inputs to transaction level using historical demand profiles.



Input: "Set Product Family A at 120,000 units in March"

Atomizer: Applies the historical profile: eg. Customer X typically takes 40% of family volume, Customer Y 35%, Customer Z 25%, with daily patterns (higher on weekdays, lower on weekends).

Output:

- March 1: Customer X, SKU-A1: 185 units.
- March 3: Customer Y, SKU-A2: 142 units.
- March 5: Customer Z, SKU-A3: 98 units.
- etc.

All aggregating back to exactly 120,000 units for Family A in March. This enables **three simultaneous capabilities**:

- Strategic flexibility (C-suite inputs high-level assumptions)
- Tactical precision (simulation executes with full customer/SKU/day detail)
- Bidirectional flow (results aggregate back up to executive dashboards instantly).

THE OPTIMISATION QUESTION

5. If simulation doesn't optimise, how do we know we're making the right choices?

This is the most common objection to simulation-based planning, and it deserves a direct answer.

5.1 When single-objective optimisation makes sense

The honest answer first: for certain well-defined problems, single-objective optimisation is the right approach.

- Tactical allocation with a clear objective (minimise cost) and stable parameters
- Well-defined constraints, frequent recurring decisions, stable cost structures
- Daily production scheduling at a single plant, where optimisation finds the best sequence in seconds

These are excellent use cases. There is no argument against optimisation here.

5.2 Why strategic decisions resist single-objective optimisation

The limitation of optimisation becomes apparent when planning decisions involve:

MULTI-OBJECTIVE REALITY WITH NO CLEAR WEIGHTING	Minimise cost vs maximise resilience vs preserve strategic optionality. These objectives shift over time, a Q1 cash conservation priority becomes a Q3 growth imperative. Optimisation requires a single objective function with defined weights. Strategic planning rarely has one.
NON-QUANTIFIABLE CONSTRAINTS	Is a unionized plant worth $X more per unit than a non-unionized one? Is Customer A's strategic relationship worth accepting $Y in margin sacrifice? These are real business constraints that cannot be expressed as coefficients.

UNKNOWABLE PARAMETERS	Vietnam labor cost 12 months out may swing ±15% with currency movements. Q4 demand could be anywhere between 70,000 and 100,000 units. Optimising with wrong assumptions produces the mathematically optimal answer to the wrong problem.

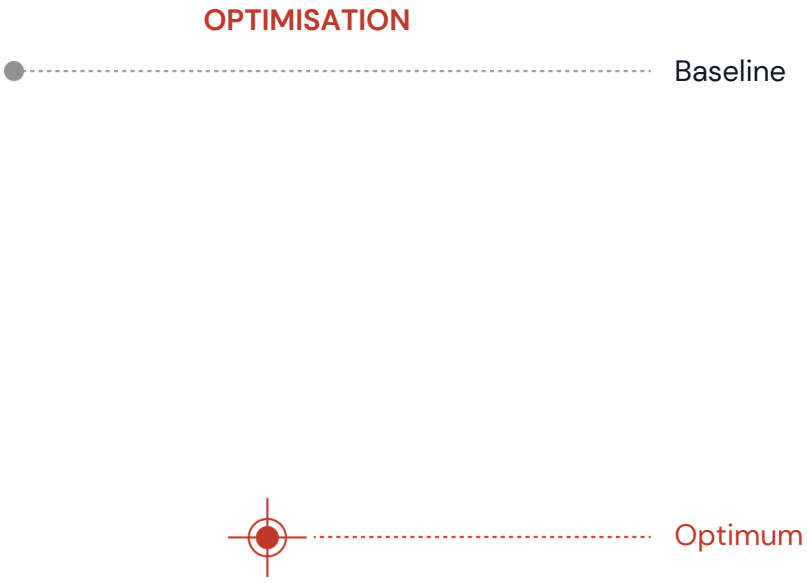
5.3 Simulation-optimisation: the best of both worlds

> **Optimisation finds a point.**
> **Simulation traces the behaviour profile that point sits on.**

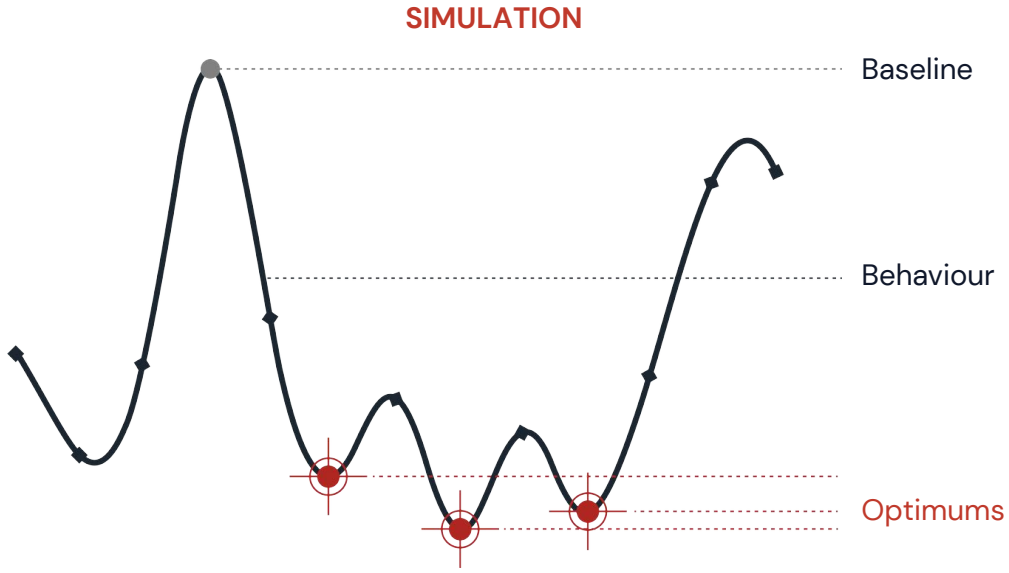
An optimiser can only return an answer once the objective and the weights between competing goals have been fixed in advance. It then hands back a single allocation. Simulation reverses the order. It runs a systematic set of scenarios through a behavioural model and traces how the business responds across the decision: revenue, profit, and working capital together. Leadership reads the trade-off directly from that profile and chooses the level that fits current priorities. The choice is grounded in the full behaviour, not asserted by a weighting no one agreed on.

This is not a claim that optimisation lacks multi-objective methods. It has them. But they require the trade-off to be committed up front, as weights or constraints, and they assume an objective smooth enough to solve. The optima that matter in practice sit on exactly the irregularities a smooth objective cannot hold: a minimum order quantity, a container fill rate, a shelf-life threshold, a supplier discount tier. Simulation finds them because it reproduces the operating rules rather than an abstraction of them.

SIMCEL's simulation-optimisation works this way. Run a systematic set of scenarios through the model, read the behaviour profile that emerges, then weigh the handful of options that are realistic. The CFO and the VP Supply Chain see the trade-offs and choose, not because the math declared an optimum, but because they understand what each option does to the business.



Returns one point. The behaviour it sits on stays hidden.



Traces the full behaviour profile. The sweet spot is read off the curve.

WHAT BUYERS THINK THEY STRUGGLE WITH

"I don't know which of a thousand options is optimal."

WHAT THEY ACTUALLY STRUGGLE WITH

"I don't understand the trade-offs, the risks, and the dynamics of the handful of options that are realistic."

Inventory policy is the clearest case, because a single decision moves three financial metrics at once and they do not move together:

- Sales: service level converting into revenue and margin.
- Profit: revenue minus every inventory-related cost, including holding, opex, and obsolescence.
- Working capital: inventory days converted into cash tied up.

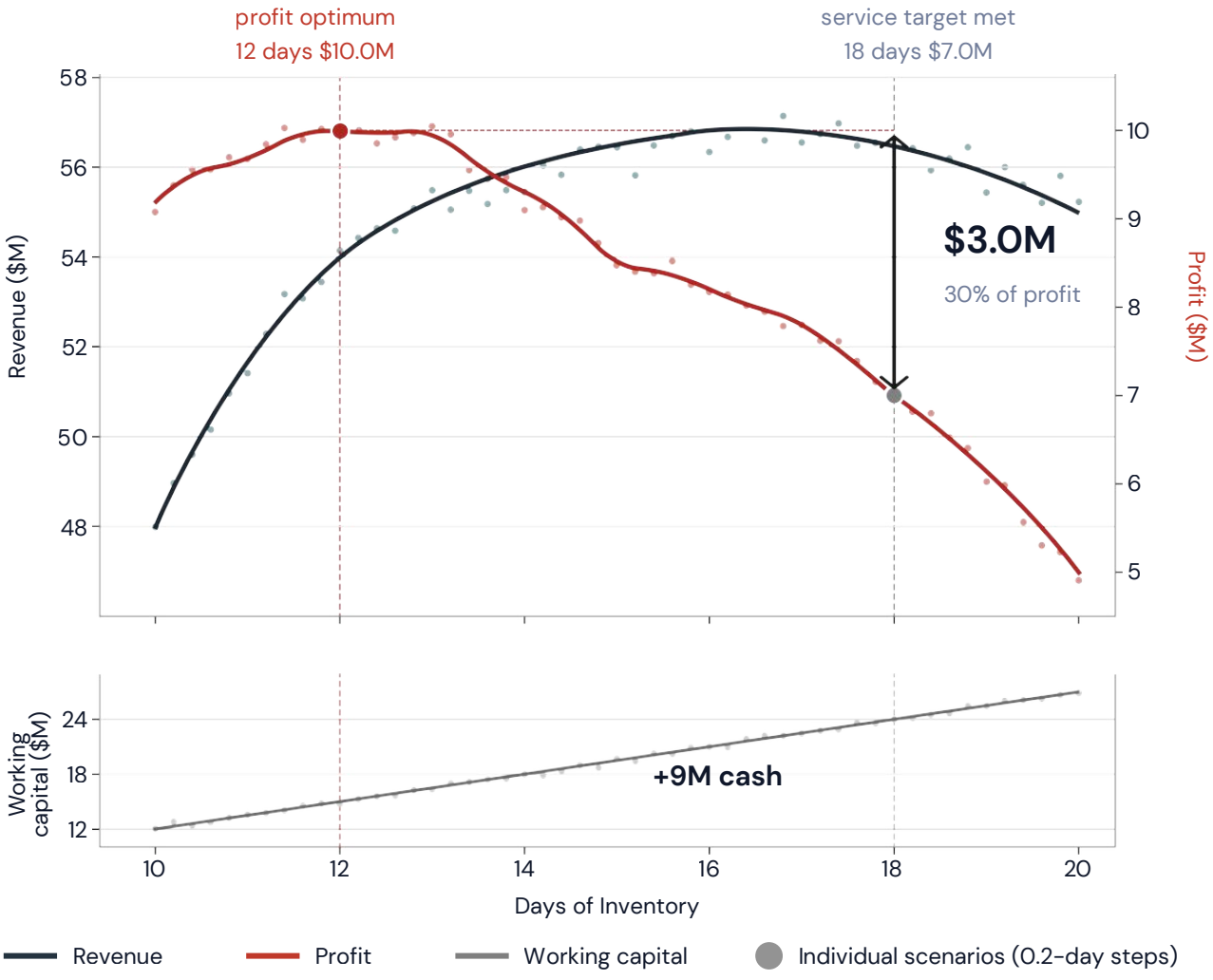
TRADITIONAL VIEW

"More inventory means higher service means more sales." Linear thinking.

SIMULATION REALITY:

Each metric follows its own non-linear profile, the three optima do not coincide, and past a point more inventory strains capacity and erodes the very service it was meant to protect. Sales themselves turn down

ONE DECISION, THREE BEHAVIOURS: THE TRADE-OFF A SINGLE OBJECTIVE HIDES



CASE STUDY: CONSUMER GOODS INVENTORY

A consumer goods company tests inventory policy from 10 to 20 days of stock, running 50 scenarios at 0.2-day increments and measuring four outcomes per scenario.

Days Stock	Service Level	Revenue	Profit	Working Capital
10	85%	$48.0M	$9.2M	$12.0M
12	92%	$54.0M	$10.0M	$15.0M
14	95%	$56.0M	$9.3M	$18.0M
16	97%	$56.8M	$8.2M	$21.0M
18	98%	$56.5M	$7.0M	$24.0M
20	97.5%	$55.0M	$5.0M	$27.0M

Four observations emerge.

1. Service follows a logarithmic curve. The first days of stock are decisive: 10 to 12 days buys seven points of service, while 16 to 18 days buys one. Diminishing returns are steep.
2. Profit peaks at 12 days, then falls hard. Below 12 days, lost sales cost more than the inventory saves. Above 12 days, holding cost, opex, and obsolescence overwhelm the shrinking revenue gain. By 18 days profit has dropped to $7.0M, and by 20 days to $5.0M, half the peak.
3. Working capital rises in a straight line. Each two additional days of stock ties up roughly $3M more cash. Moving from the profit optimum at 12 days to the service-driven plan at 18 locks up $9M in additional inventory.
4. Past their peaks, both service and the sales it converts reverse. Revenue peaks near 16 days at $56.8M, then eases to $56.5M at 18 days and $55.0M at 20. Service holds up a little longer, reaching 98% at 18 days before slipping to 97.5% at 20. Beyond roughly 18 days the inventory meant to protect service begins to undermine it: storage fills, handling slows, stock ages. The intuition that more inventory always buys more sales does not merely flatten here, it inverts, and a linear model cannot produce that turn at all. It is the clearest refutation of the more-stock-more-sales assumption.

What a single-objective optimiser recommends

The company's availability policy targets 98% service. A cost-minimizing solver meets it at the lowest inventory that satisfies the constraint: 18 days. It reports success, 98% service achieved at minimum cost, and stops. It cannot see what simulation shows on the same data: 18 days delivers $7.0M profit, while 12 days delivers $10.0M. The service target, enforced blindly, costs $3.0M, 30% of profit, and ties up $9M more cash, to buy six points of service the business never priced. The optimiser is not wrong about service. It is blind to profit, because service is not profit, and its linear cost model cannot see the holding, obsolescence, and capacity dynamics that bend the profit curve.

> **The service target, enforced blindly, costs 30% of profit, to buy six points of service the business never priced.**

Threshold effects: why the curve is not smooth

The table samples every two days; the simulation runs every 0.2 days, and at that resolution the profit curve in Chart B is visibly not a smooth slope. Four operating rules interrupt it, each one invisible to a linear cost model and each falling on the odd days between the table's rows.

- 11 days, profit dip. SKU-A crosses its minimum order quantity. Replenishment forces a 5,000-unit run against the 3,200 actually needed, and the 1,800 excess units sit in stock. The dip just left of the 12-day peak.
- 13 days, profit spike. Inventory is now high enough to fill 40-foot containers above 95%. Freight consolidates and unit shipping cost drops from $12 to $8. The lift early on the descent.
- 15 days, profit dip. SKU-B crosses a 60-day age threshold. Shelf-life policy triggers a 20% markdown and raises obsolescence risk. The deepest notch on the slope.
- 17 days, improvement. Volume clears a supplier discount tier, a 5% reduction on bulk purchasing. The small late uptick before the decline resumes.

A linear optimiser cannot produce any of these. Fixed per-unit coefficients smooth a batch quantity, a container fill rate, an age limit, and a discount tier into one straight line, and the texture disappears, along with the optima and the traps that live on it.

What simulation-optimisation delivers: a priced menu.

Instead of a single optimal answer, the CFO and VP Supply Chain see a priced menu of defensible options and choose by priority:

10 days
BEST CASH POSITION

- Working capital $12M
- Profit $9.2M
- Service 85%

12 DAYS
BEST PROFIT

- Working capital $15M
- Profit $10.0M
- Service 92%

16 days
BEST SERVICE

- Working capital $21M
- Profit $8.2M
- Service 97%

The menu stops at 16 days, not 18, by design. The last point of service, 98% at 18 days against 97% at 16, costs $1.2M of profit and $3M more cash to buy. That is the option a cost-minimising solver, told to hold 98%, would have selected, and the one this analysis has already priced as not worth paying for. It is left off the menu deliberately.

Each option is now legible: leadership can see that the last points of service cost real profit and cash, and decide whether the business wants to pay for them this quarter. On a Q2 profit-and-cash priority the recommendation is 12 days, shifting toward 16 for a Q4 peak-season service push. An informed choice with the full price of each option visible, not a black-box optimum. Fifty scenarios, 15 to 20 minutes.

THE ARCHITECTURAL CHOICE

6. Which approach fits which problem

The choice between these architectures is usually framed as a contest between two planning tools. It is more useful as a question about the conditions a business plans under, and what kind of system those conditions require.

Optimisation-based planning was built for a world that holds still long enough to be solved: stable costs, known parameters, a single agreed objective. That world has not vanished, but it has narrowed. Even supply chains whose physical structure is unchanged now absorb continuous cost and price movement, in inputs, freight, energy, and currency, and that movement is not a phase to wait out. It has become the operating condition, and it is not reverting to quiet seas.

This does not make optimisation wrong. It changes where it fits. The slice of the planning problem that stays still long enough to optimise has shrunk, and the part optimisation was never designed for, navigating uncertainty as it unfolds and deciding what to do when the parameters move, is now where most of the value and most of the risk sit. The missing piece is not a faster solver. It is a layer that lets leadership see how the business behaves under conditions it cannot predict, before committing to a plan.

That is the real architectural choice: not which tool is better, but which kind of artifact the problem in front of you requires.

6.1 Two kinds of artifact

The two architectures are not two versions of the same product. They are different kinds of things:

OPTIMISATION-BASED PLANNING: AN ANSWER MACHINE

Commit a model, constraints, and an objective, and it returns the allocation that satisfies them. You execute the number it returns.

MULTI-PARADIGM SIMULATION: A BEHAVIOURAL MODEL

A decision environment you interrogate. It shows how the business behaves across the choices in front of you, and leadership decides.

The relationship between them is asymmetric, and the asymmetry settles which is the more fundamental. You can run an optimisation inside a simulation: that is exactly what the inventory case in Section 5.3 did, sweeping a parameter and reading the behaviour. You cannot run a simulation inside an optimiser. Optimisation is a technique. The behavioural model is the representation of reality that techniques run inside. This is why simulation is not a lower-level execution layer beneath the optimiser; it is the layer that contains it.

> **You can run an optimisation inside a simulation. You cannot run a simulation inside an optimiser**

The rows below are where those five architectural properties show up in day-to-day planning.

AN ANSWER MACHINE VERSUS A BEHAVIOURAL MODEL

IBP Requirement	Graph-based optimisation	Multi-paradigm simulation
Demand-Supply Balancing	Excels (native capability)	Excels (SD + ABM Layers)
Financial Integration	Requires separate FP&A tool + reconciliation	Native (P&L emerges from simulation)
Working Capital Precision	Approximation (average days)	Transaction-level timing (actual cash flow)
Cross-functional Alignment	Manual (Sales, Ops, Finance use different tools)	Automatic (one model, one truth)
Plan Consistency	Requires 2-3 week reconciliation cycles	Inherent (same simulation produces all outputs)
Changing Cost or Demand Conditions	Optimisation re-solves per change, can be stale before execution	Simulation re-runs natively as conditions move
Resilience and Stress Testing	Optimisation a structural limitation	Simulation native
LARGE-SCALE VOLUME-ALLOCATION SPEED AND SCALE	Excels at fast large-scale allocation	Can model allocation, but heavier for pure optimisation

Native capability Partial/workaround required Structural limitation

6.2 When graph-based optimisation is the right choice

The honesty this paper has held throughout requires stating plainly where graph optimisation is the right architecture, and it genuinely is when its conditions hold:

- Large-scale volume allocation across a global network with a single clear objective.
- Stable, linear cost structures, commodities, fixed logistics rates, predictable labor,
- where a per-unit coefficient is a fair approximation.
- Monthly planning cycles where aggregation to families and segments does not compromise the decision.
- Strategic network design, plant and distribution-centre footprints, where structure matters more than dynamics.

The common thread is stability.

> **Optimisation is the right architecture when the world holds still long enough for a single optimal answer to remain true between the moment it is computed and the moment it is executed.**

Genuinely stable pockets do exist, regulated utilities, true commodities, mature staples, and this paper does not pretend otherwise. Two cautions apply even there. Structural stability is not cost stability: a supply chain whose flows never change can still be shaken by input-cost and price swings, and the linear-coefficient assumption is the first thing those swings break. And the set of businesses that can treat their planning environment as stable is contracting, not growing.

6.3 When multi-paradigm simulation is the right choice

Multi-paradigm simulation is the right architecture when the conditions optimisation needs do not hold, which is increasingly the common case:

- Volatility, in demand, cost, or supply, is high enough that a single optimal answer is stale before it is executed.
- Financial precision matters at the SKU and customer level: absorption accounting, working-capital timing, FX, revenue recognition against cash collection. The same transaction stream carries any other dimension at the same precision, carbon included: actual and projected emissions at SKU, lane, and facility level, computed in the same model as cost and cash, where an optimiser can only attach an aggregated coefficient.

- The decision spans objectives, service, profit, cash, resilience, that cannot honestly be reduced to one weighted function.
- Consistency across horizons matters. The plan must hold from weekly execution to quarterly strategy without a fidelity break. Simulation keeps one granularity and one financial truth at every cadence, because it aggregates from actual transactions rather than planning at an aggregated level; optimisation has a cadence sweet spot precisely because aggregation is its precondition, not a choice.
- Autonomous, AI-assisted planning is on the roadmap, which a behavioural model can support and a periodic re-optimiser cannot.

Underneath this list is the capability the current environment actually demands: agility and resilience. Neither is a feature that can be bolted onto an optimiser. They are properties of being able to ask what happens if against a model of how the business behaves, and to see the answer before committing. An optimiser re-solves for new parameters; it cannot show how the system absorbs a shock, where the strain accumulates, or how long recovery takes, because those are behaviours over time, not allocations at a point.

This is most concrete in stress testing. A behavioural model can be subjected to the shocks a business fears, a lost supplier, a demand spike, a cost or currency jump, to quantify what each does to service, profit, and cash, and to test the responses before they are needed. For risk-exposed businesses this is the decisive difference: discovering a plan's breaking point in a simulation rather than in the market, and arriving at the disruption with the response already rehearsed.

> **Simulation does not create resilience. It makes resilience visible and testable before the decision is made.**

That boundary keeps the claim defensible. The decision still belongs to the leadership team; simulation's role is to show what each option does before the team commits, the same see-the-trade-off-then-choose posture as Section 5.3.

Representative implementations are not a specialist niche but the default for businesses exposed to volatility and tight financial coupling: pharmaceutical planning with SKU-level shelf-life and customer-specific service terms, consumer-goods promotional planning with multi-channel margin precision, high-tech supply chains with volatile demand, and private-equity portfolio companies where financial precision is the mandate.

6.4 Integration where the network is truly global

Organisations with mature optimisation investments do not face a binary choice, but the right relationship is not a hybrid in the literal sense. Running two planning platforms side by side, each owning part of the model, is technically complex and tends to recreate the two-tool reconciliation problem this series exists to solve. That is rarely worth it.

A cleaner integration is directional and geographic rather than strategic-versus-tactical. It runs as a loop:

- On the demand side, simulation predicts what each market will require, converting country-level demand and distribution behaviour into import requirements.
- Global supply is then allocated by optimisation, which is genuinely strong here: many sources, shared capacity, one cost objective, across a truly global network.
- That allocation returns to simulation as constrained supply, and the behavioural model reverts it to each country as a constrained distribution and sales plan: what can actually reach demand, at what cost, and at what working capital.

DIRECTIONAL-INTEGRATION LOOP. SIMULATION OWNS THE DEMAND AND EXECUTION ENDS; OPTIMISATION SOLVES GLOBAL ALLOCATION IN THE MIDDLE.



Neither system sits under the other. The optimiser answers one question: how should constrained global supply be allocated. Simulation answers the questions on either side of it: what the market will require coming in, and what the business will actually do going out. Simulation touches reality on both ends, demand sensing and execution; optimisation is the specialized solver invoked in the middle for the one problem it is best at.

At regional level the balance tilts further toward simulation. Distribution faces more external variation than upstream allocation does, and coupling financial and supply-chain performance at the regional level is where measurable gains tend to sit.

A word on dependency, because both architectures create it. The honest difference is not that simulation avoids lock-in; a behavioural model demands transaction-level data and modelling discipline, which is its own commitment. The difference is the nature of the dependency. An optimiser's output is a number to be trusted, and when it diverges from reality, which this paper has argued it will, the abstraction makes the divergence hard to trace, and trust leaks into the shadow spreadsheets that fill the gap. A behavioural model is inspectable: you can watch the business behave, which builds judgment rather than blind reliance. The dependency is on a transparent model tied to actual transactions, not on an abstraction that drifts from them.

A single platform is the better choice for greenfield IBP implementations, mid-market organisations without heavy legacy investment, organisations worn down by two-tool reconciliation, and any organisation for which AI-enabled autonomous planning is a strategic priority. For these, a unified, simulation-native architecture is simpler to run and removes the reconciliation burden at the root rather than managing it across an integration.

CONCLUSION

The question this paper answers

Optimisation-based planning and multi-paradigm simulation are not competing answers to the same question. They are different kinds of system, built for different conditions, and the organisations that read the difference correctly make better infrastructure decisions than those who treat it as a feature comparison.

Optimisation-based planning has earned its place across two decades of enterprise planning. Its precision on large-scale allocation, its speed on monthly volume cycles, and the clarity it brings to a complex network make it the right architecture for the problem it was built to solve: stable costs, known parameters, a single agreed objective, a world that holds still long enough to be solved.

Multi-paradigm simulation answers the questions on either side of that allocation: what the business will actually require, and what it will actually do once the plan meets time, variability, and the operating rules a coefficient cannot hold. It is the more fundamental of the two, and not by assertion. You can run an optimisation inside a simulation; you cannot run a simulation inside an optimiser. That asymmetry settles the order between them: behaviour is the layer, and allocation is a technique that runs within it.

Paper 1 asked whether the gap between planning and finance was a process problem or an architectural one. This paper answered the architectural question: what the two approaches compute, where each is genuinely right, and why the boundary between them is fixed by the decision layer rather than by configuration or implementation quality. Paper 3 takes the next step, into what happens to a plan that is sound at the monthly level when it crosses into weekly execution, and why that altitude break, not the solver, is where revenue and margin quietly leak.

The question this paper set out to answer is the one a planner meets every time the system's number and the business's outcome part ways, and the gap is closed by hand in a spreadsheet no one fully trusts: is that gap a solver problem, a configuration problem, or an architectural one?

Where it is occasional, it is none of these; the system is fine. Where it is persistent and material, it is architectural. Your plan is optimal, and your business disagrees, because the plan was solved for and the business was not. No solver setting closes that gap; the limitation is not in how well the system solves, but in what it was built to compute.

The opening question

The next time your planning system produces a scenario and you find yourself asking

> ***"But what actually happens between now and then?"***

ASK:

IS MY SYSTEM MODELLING BEHAVIOUR, OR FINDING ALLOCATIONS?

The answer will tell you which architectural questions are worth asking next.

ABOUT THIS SERIES

One model, One truth

7.1 About the series

This paper is part of a five-paper series examining the architectural foundations of Integrated Business Planning and the implications for financial precision, operational fidelity, and AI-enabled decision intelligence.

PAPER 1

Escaping the IBP Reconciliation Trap

PAPER 2 (THIS PAPER)

Optimal, and Still Wrong

PAPER 3

The Fidelity Problem: Why Strategic Plans Break Down in Tactical Execution

PAPER 4

The Hidden Cost of Standard Costs: The Financial Imprecision Your CFO Can't See

PAPER 5

The Path to Autonomous Planning: Why True AI Requires a Simulation-Native Foundation

7.2 About SIMCEL

When sales, supply chain, and finance plan in disconnected silos, strategy separates from execution, leading to slow responses, hidden profit leaks, and missed revenue.

SIMCEL solves this as an Enterprise SaaS platform that integrates demand, supply, and financial planning into a single simulation-based decision engine.

Instead of juggling static spreadsheets, leadership teams can test end-to-end business scenarios and see the exact financial impact from operational execution straight through to the full P&L.

Because SIMCEL simulates actual SKU flows to customers rather than relying on high-level averages, organisations can instantly evaluate critical trade-offs like revenue versus profit margins, or cash flow versus capacity.

Teams can run a complex 'what-if' scenario, such as a supplier failure, a portfolio rationalisation, or a sudden demand shift, in under 60 seconds, and a full scenario sweep in minutes.

By providing one source of truth, SIMCEL removes version conflicts and cuts planning cycle time by 50 to 75% across its deployments. The engine behind it has a longer track record. In CEL advisory engagements, before it became a platform, it optimised inventory and service policy for profit rather than cost. Across more than nine such engagements, each with a documented baseline, it reduced cost-to-serve, the logistics, distribution, and service spend that sits between revenue and profit, by roughly 9% on average and by close to 20% where a client's starting policy sat furthest from its profit-optimal point. Where the impact was measured at the profit line directly, it ran 7 to 10%, and higher still in cases where the work also resolved a capacity constraint, which many businesses face.

SIMCEL

OPTIMAL,

AND STILL WRONG

Discover how SIMCEL's multi-paradigm simulation architecture models the full dynamic behaviour of your business demand, supply, and finance; in a single unified model.

BOOK YOUR DEMO

info@simcel.io

© 2026 SIMCEL Pte. Ltd. All rights reserved.