

How to Optimise a Company. It Depends.

UNDERSTAND THE SYSTEM
NOT JUST THE KPI



AURELIX
SYSTEMS

Mateusz Chrzanowski

SYSTEM OF SYSTEMS ARCHITECT



THE QUESTION / 00

"How would you improve our company?"

I don't know.

THE ONLY PROFESSIONAL ANSWER BEFORE UNDERSTANDING THE SYSTEM

That is the only professional answer I can give before understanding the company. Not because I lack methods, experience or technology, but because I have not yet established what I am supposed to improve. Anyone who already knows the answer before understanding the system knows the answer to a different question.

Some consultants will look at your company's website, read a few pages about what you do, and confidently promise to improve your profitability by 20 per cent. They haven't examined your operations, reconstructed your dependencies or calculated the consequences of a single proposed change. They know what your company says about itself. They know practically nothing about how the company actually works.

Yet somehow, they already know the percentage.

Apparently, your entire enterprise architecture, together with its future financial performance, was included on the About Us page.

I find that fascinating.

There is nothing wrong with knowing how to improve a process. A manufacturer of ceramic rollers may know perfectly well that its product can reduce wear, improve operating conditions or extend maintenance intervals. An experienced engineer can identify a technical deficiency and propose a better component. An organisational specialist may recognise a badly designed workflow almost immediately. None of those people has to pretend that they know nothing.

But knowing that a solution improves a particular function does not establish its value to a particular company. It does not tell us how much money the company will make, how much it will save, what new dependencies it will create or whether the proposed intervention is even the best use of the company's capital.

And it certainly does not establish the optimum of the entire enterprise.

A solution can be technically excellent and economically inappropriate. A process can become substantially faster without increasing company profit by a single euro. An investment can improve resilience, extend the useful life of equipment or prevent a future operational failure without producing an impressive quarterly financial result.

That is still improvement.

The mistake is assuming that every improvement can be expressed as a universal percentage and that every percentage represents the same thing.

Before we start discussing the best way to optimise a company, we need to establish what a company actually is.

THE COMPANY ON PAPER DOES NOT EXIST

Most management boards have access to an extraordinary quantity of information. Revenue, EBITDA, operating margins, labour costs, machine utilisation, inventory, production volumes, delivery performance, employee turnover and hundreds of other indicators can be displayed on beautifully designed dashboards.

Every number may be mathematically correct.

That does not mean management understands the company.

Imagine a dashboard showing that warehouse performance has fallen below target. The indicator turns red, management identifies a problem and somebody receives an instruction to optimise the warehouse. Perhaps the proposed answer is better inventory management, additional automation, a new WMS or a different storage arrangement.

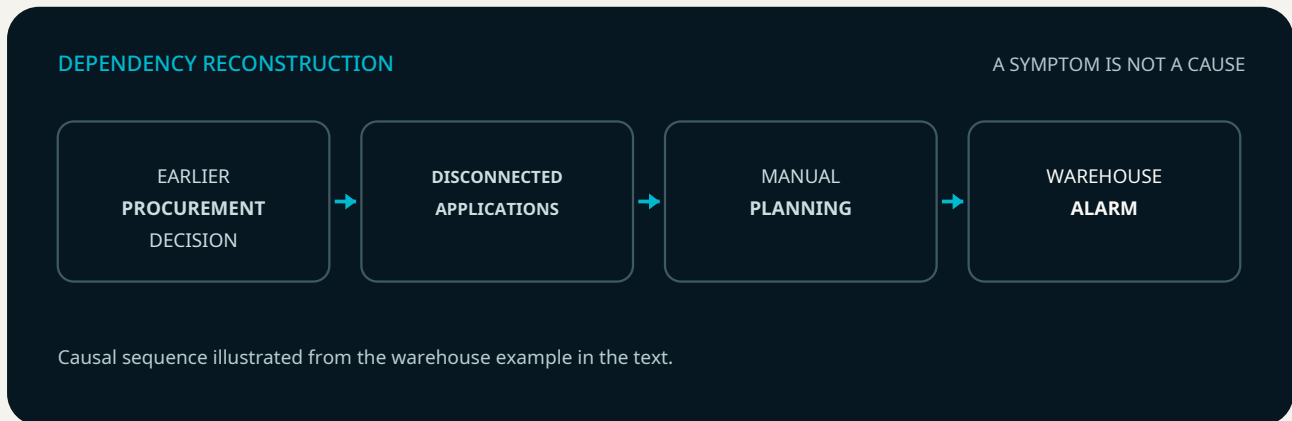
All perfectly plausible solutions.

But the warehouse may have no architectural problem of its own.

Its performance may have deteriorated because production is being scheduled in the wrong sequence. The production schedule may be wrong because a planner is combining information manually from several applications. Those applications may not communicate because a procurement decision made years earlier introduced an interface that never provided the required information. The warehouse is simply the first place where the consequences become sufficiently visible to trigger an alarm.

You can spend a fortune optimising that warehouse without correcting the actual cause. You may even make the entire system worse by removing the capacity that previously absorbed the disturbance.

A dashboard can tell you where a symptom has appeared. It cannot automatically tell you where the problem originated.



Data describes the result. Architecture explains how the result is produced.

This distinction matters because a company is an interconnected system. Its employees, equipment, software, suppliers, customers, physical infrastructure, processes, finances and management decisions interact continuously. A change in one part of the organisation can produce consequences somewhere completely different, sometimes after a considerable delay.

You can reduce inventory by transferring shortages into production. You can reduce labour costs by transferring work onto fewer employees. You can increase machine utilisation by consuming maintenance capacity. Every individual indicator may move in the desired direction while the enterprise becomes progressively less stable.

An organisation chart describes formal responsibility, not necessarily the actual movement of decisions. A procedure describes how work is supposed to happen, not necessarily how people perform it. A schedule describes an intended sequence, not the physical capacity to execute it. A financial report describes the results recognised during a particular period, not necessarily the operational conditions that produced them.

These are different things.

If we want to optimise the enterprise rather than improve an isolated indicator, we have to understand the relationships between them.

And there is another principle that makes the entire problem considerably more interesting.

EVERY EFFECT HAS A COST

Every effect has a cost.

I mean that literally.

Any effect we want to achieve requires a particular combination of resources, capabilities, operating conditions and dependencies. Producing that effect consumes something. Sustaining it consumes something. Changing the system to make it possible may require an additional investment.

Until we have identified and calculated the relevant variables, we do not know the true cost of the effect.

This does not mean every improvement is dangerous or that every increase in production carries some mysterious penalty. It means we have to stop confusing the visible cost of an action with the total cost of the result we want.

Suppose a production machine normally operates at 60 per cent of its available performance. Management discovers that it can operate at 80 per cent. The increase looks attractive, particularly when demand is sufficient to sell everything the factory can produce.

The machine can physically run faster.

Excellent.

That tells us the desired output is technically achievable. It does not establish the mechanical cost of sustaining it.

Consider a Škoda Fabia with a 1.0-litre TSI engine and a Porsche 911 Carrera 4S travelling at 180 km/h.

Both cars are travelling at exactly the same speed.

The Porsche is operating at its optimum. Its engine and drivetrain are designed to sustain those conditions without sacrificing their expected service life. The Fabia, meanwhile, is operating practically at its maximum. Sustaining that performance places extraordinary demands on its mechanical components, dramatically accelerating wear and potentially leading to serious failures in a comparatively short time.

Same speed. Two completely different mechanical realities.

And industrial machinery obeys exactly the same laws of physics.

Mechanics does not negotiate with management.

Bearings provide an excellent example. Their fatigue life does not decrease linearly with increasing load: for ball bearings, the basic load-life relationship follows a third-power law. An increase in actual bearing load can therefore produce a disproportionately large reduction in expected fatigue life. Unbalanced rollers introduce another problem, because the dynamic forces generated by imbalance rise with the square of rotational speed. What appears to be a minor imperfection at lower speeds can become a serious source of vibration and accelerated wear. Higher contact loads and temperatures can also destroy the lubricating film between gear teeth, allowing microscopic adhesive junctions to form and tear material from the surfaces as they separate.

And those are just three examples.

A production machine contains hundreds of interacting mechanical elements, each with its own operating conditions, accumulated wear and failure mechanisms.

This is engineering, not management philosophy.

Perhaps the machine needs ceramic bearings. Perhaps conventional rollers should be replaced with elements using ceramic surfaces deposited by chemical vapour deposition. Maybe a suitable diamond-like carbon coating could improve the tribological behaviour of particular components. Better balancing, gear geometry, surface treatment, lubrication or cooling might be necessary.

Which solution is correct?

It depends.

Ceramic bearings are not automatically better bearings. CVD coatings are not magic, and DLC does not improve every mechanical interface simply because it is an advanced material. Their usefulness depends on loads, temperatures, contact conditions, lubrication, substrate properties, the failure mechanism and the required operating life.

We must understand the machine before specifying the modification.

Now expand the analysis beyond the machine itself. Increasing its output changes the amount of material entering and leaving the workstation. Operators may have more tasks to perform, quality control may need additional capacity, maintenance may require a different schedule, and warehouse logistics may have to handle substantially greater throughput. The planning system needs to maintain the correct material sequence. Purchasing needs to deliver sufficient material. Internal transport must support the new flow. Documentation, supervision and information systems have to remain capable of controlling the process.

The machine may be perfectly capable of producing more.

The factory may not be.

That is why increasing the operating speed of one machine is not necessarily the same thing as increasing the capability of a production system.

Let us put some money into the problem.

Assume the increased production volume would generate an additional €5 million of annual operating contribution after the directly associated production costs, including additional raw material. I am going to assume that management has already remembered to calculate the material. That would be a pleasant surprise.

We analyse the system and discover that achieving this increase sustainably requires a series of changes. Modifying the machine will cost €200,000. Rebuilding the operator workstations will require another €500,000. A new production supervision system will cost €1.5 million. The warehouse needs €1 million of modifications, while improvements to office processes and information flow require another €500,000.

The total investment is €3.7 million.

Round it up to €4 million.

The new operating model also requires higher expenditure. Additional employee remuneration and bonuses amount to €1.5 million annually, and additional personnel cost another €500,000. The annual increase in operating expenditure is therefore €2 million. We also allocate €500,000 a year as a conservative allowance for additional maintenance and operating requirements.

We started with an additional €5 million of annual operating contribution. After the €2 million increase in OPEX and the €500,000 operating allowance, we retain approximately €2.5 million of additional annual benefit.

Against €4 million of initial investment, the simple payback period is about 1.6 years.

SYSTEM-LEVEL INVESTMENT / ILLUSTRATIVE MODEL

€5.0M

ADDED CONTRIBUTION

€2.0M

ANNUAL OPEX

€0.5M

OPERATING ALLOWANCE

€2.5M

NET BENEFIT / YEAR

€4.0M

INITIAL CAPEX

~1.6 YEARS

SIMPLE PAYBACK

Figures follow the hypothetical production scenario in the article.

That is a very attractive result.

We should seriously consider making the investment.

And notice what has happened.

We have increased expenditure, purchased more expensive components, improved operator conditions, invested in software, modified the warehouse and employed additional people. Our annual operating costs are higher.

Yet the company is better.

We have not merely pushed one machine harder. We have increased the capability of the system surrounding it and established an economically attractive operating point.

There is also a financial distinction that must remain clear. The €4 million of CAPEX is the initial investment required to establish that capability. It does not reduce the €2.5 million of recurring annual operating improvement. Financing, taxation, depreciation and the time value of money belong in the full investment evaluation, but they do not turn a one-off investment into an annual operating expense.

After the modifications, we may even discover that the original 80 per cent target was conservative. The upgraded machine could potentially achieve the equivalent of its previous 90 per cent output without the operating conditions that would once have made that level uneconomical.

Whether it can do so is another engineering question.

And another calculation.

The point is that optimisation does not mean reducing the cost of every component. It means designing the system to achieve the required result at the best total cost, subject to the constraints that actually matter.

Now consider what happens if management skips that analysis.

Instead of investing €4 million, the company simply increases machine speed and tells everyone to work harder.

Output improves immediately. The first quarter looks excellent, the second is even better, and by the fourth quarter the dashboard is no longer green.

It is practically gold.

Production has increased. Revenue has increased. The financial results appear to confirm that the decision was brilliant. Someone presents the figures to the board and receives a substantial bonus.

They open the champagne.

And drink it with money borrowed from the future.

Then reality begins to appear.

The machine requires €1.5 million of additional servicing after a year of harder operation. Components begin failing more frequently. Maintenance interruptions increase. Quality problems produce more rejected material and more customer complaints. Employee turnover rises because the original workforce has been operating under increased pressure without any corresponding improvement in the working environment.

The nominal 80 per cent output may become an effective 70 per cent after additional downtime, defects and production losses. The remaining useful life of expensive machinery may have shortened considerably, moving replacement expenditure forward by several years.

After recalculating the actual costs, the company may discover that the supposed improvement generated little additional profit.

It may even have produced a net loss.

The original decision looked excellent because its benefit appeared immediately while a significant part of its cost accumulated over time.

The first four quarters showed the effect.

They did not show its full cost.

And the champagne has already been drunk.

This is one of the most dangerous characteristics of complex systems. The time at which a decision produces a measurable benefit does not have to coincide with the time at which its consequences become measurable.

A machine can continue operating while consuming its remaining useful life faster than expected. An employee can maintain the required output while progressively losing the attention, tolerance and energy needed to sustain that performance. A warehouse can handle additional material while consuming the spare space that previously allowed it to absorb disturbances.

Nothing has failed.

The required output is still being produced.

Management concludes that everything works.

But “it works” is not a measure of system quality.

It only means that the required output is still being produced.

SECTION 03

ENTERPRISE / SYSTEM ARCHITECTURE

THIRTY SECONDS

Sometimes the opposite happens. A change is so small that nobody considers it financially significant.

We take an operator performing a repeated task. We redesign the workstation, change the position of a tool and eliminate an unnecessary movement.

We recover thirty seconds from each operation.

Thirty seconds.

Hardly an achievement worth announcing at the next board meeting.

Now that operation occurs ten times per hour. We have recovered five minutes of working capacity per hour on one workstation. The factory operates twenty identical stations across a continuous twenty-four-hour schedule.

That is 100 minutes of released working capacity every hour across the factory, or forty working hours every day.

Over 360 operating days, the change releases 14,400 working hours annually.

At a fully loaded labour cost of €20 per hour, that capacity has a labour-cost equivalent of €288,000 per year.

SMALL OPERATIONAL CHANGES / SYSTEM SCALE

30s

PER OPERATION

20

WORKSTATIONS

14,400h

CAPACITY / YEAR

€288k

LABOUR-COST EQUIVALENT

Released capacity is not automatically cash saved in wages.

Does the company save €288,000 in wages?

Of course not.

The employees still work. They are still paid. Nobody's salary has been reduced simply because the process became faster.

What changed is the amount of productive work that can be performed using the same labour resources.

Now we use those forty working hours to produce another 400 finished products every day. We sell those products with €100 of contribution from each unit after the relevant variable costs.

That gives us €40,000 every day.

€14.4 million annually.

Thirty seconds.

And now comes the question that actually matters.

Can the company produce and sell those additional 400 units every day?

Are the machines capable of handling the additional volume? Are enough materials available? Can quality control and logistics keep up? Is another workstation already limiting production? Does the market have sufficient demand?

In one factory, this improvement creates millions in additional recurring value.

In another, output does not increase at all because a different part of the system has already reached its limit.

In a third, the principal benefit may be reduced fatigue, lower overtime, better quality or greater operating resilience.

The workstation has improved in every case.

The financial consequences are completely different.

And we cannot simply add the €288,000 labour-cost equivalent to the €14.4 million as another saving. We are describing two ways of valuing the same released capacity, not discovering two independent sources of benefit.

That is precisely why understanding the difference matters.

The method worked. The workstation improved. The actual value to the enterprise still depends on the enterprise.

THE COMPANY CAN CONSUME CAPITAL IT DOES NOT RECOGNISE

There are resources inside an organisation that are exceptionally valuable but rarely appear as recognisable assets on a balance sheet.

Spare capacity is one. Human attention is another. There is also process tolerance, management bandwidth, the remaining useful life of equipment, accumulated employee knowledge, supplier relationships, organisational resilience and the ability to absorb unforeseen events.

These resources form a kind of invisible operational capital.

They allow a company to function when reality does not follow the plan. A production line with sufficient spare capacity can absorb a temporary increase in demand. A maintenance department with available people can respond to an unexpected failure. A warehouse with free space can handle an abnormal delivery. An experienced employee with sufficient time can identify a problem before it propagates through several departments.

None of these reserves is automatically waste.

Sometimes spare capacity is unnecessary. Sometimes additional inventory is excessive. Sometimes a process genuinely contains redundant activities. Those are legitimate opportunities for improvement.

But spare capacity can also be an essential part of the architecture.

Remove every buffer, increase every utilisation indicator and reduce every apparently unnecessary expense, and the result may be a company that performs wonderfully under ideal conditions but becomes unstable as soon as anything unexpected happens.

The interesting part is that this deterioration may take years to become visible.

Imagine that the increased workload in our factory causes operators to perform slightly more actions every hour. Initially, nobody complains. After several months, small errors become more frequent. After a year, experienced employees are increasingly exhausted. Two years later, several of them leave.

The original production decision was made in Year 1.

The turnover appears in Year 3.

By then the connection may have disappeared from the organisation's understanding of itself. Management sees recruitment difficulties, an expensive labour market or insufficient employee engagement. Human resources receives a new problem to solve.

Nobody necessarily associates the turnover with the production target introduced two years earlier.

The cost has moved through time and through the organisation.

And the same problem appears in purchasing.

Suppose a company avoids spending €100,000. The immediate financial result is clear: €100,000 has not left the account. Someone may be tempted to report a saving.

But not spending money is not the same as saving money.

If the expenditure was unnecessary, eliminating it may genuinely improve the company. If avoiding it increases maintenance, reduces quality, consumes employee time or creates a much more expensive downstream process, the result is different.

Fifteen proper cable rollers have an easily identifiable purchase price. Their absence may have a cost distributed across the working hours of several electricians, longer installation times, damaged cable insulation and increased physical effort.

A procurement spreadsheet can recognise the invoice immediately.

It may never identify the work created by refusing the purchase.

A company may also avoid an expenditure by failing to pay a subcontractor properly. The dispute might never reach court, and the unpaid amount could remain in the bank. But that does not establish a system saving.

The supplier may refuse future work. Other contractors may increase prices to cover the perceived risk. The organisation may lose reliable external capacity that is difficult to replace.

The consequence does not have to return as the original invoice.

It may reappear as a completely different operational or financial problem, perhaps much later.

A reduction in expenditure is not a saving until its effect on the system has been verified.

Now imagine an owner deliberately exploiting this timing problem.

For three years, the company concentrates on making its financial indicators as attractive as possible. Maintenance is postponed, investment is delayed, staffing is reduced, working capital is squeezed, equipment is operated harder and the organisation's remaining reserves are gradually consumed.

The financial results improve. EBITDA looks impressive, utilisation is excellent and the company appears exceptionally efficient.

Then the owner sells it.

The seller may never personally experience the full consequences. The buyer receives the company together with its equipment, employees, suppliers, processes and accumulated operational debt.

That debt may consist of deferred maintenance, exhausted staff, underdeveloped systems, equipment approaching failure, damaged supplier relationships and a lack of capacity to absorb the next disturbance.

Much of it may be difficult to quantify during conventional due diligence, particularly when historical performance appears strong.

The business was not optimised for sale.

It was prepared for sale.

There is a difference.

A company can report profit while consuming capital that it does not recognise as capital. If the consumption is sufficiently gradual, the organisation may operate for years before the consequences become visible.

And when they finally appear, nobody can guarantee that the cost will be proportional to the original saving.

This is system debt.

Unlike conventional financial debt, it does not necessarily come with an identifiable creditor, interest rate or repayment schedule.

It may remain hidden until the operating conditions change.

THE COST OF A DECISION CAN ARRIVE YEARS LATER

The principle is the same at a much larger industrial scale.

Consider the 1.2-litre PureTech petrol engine and its oil-immersed timing belt. Earlier generations of these engines became associated with premature belt degradation and other significant reliability problems. Stellantis subsequently introduced extended warranty protection and compensation arrangements for qualifying owners.

The engineering risks of an arrangement like this have to be evaluated during development. Materials, lubrication conditions, operating temperatures, ageing behaviour and the consequences of belt degradation all belong to that evaluation.

The exact internal economic reasoning behind the original design belongs to the manufacturer.

But the economic attraction of small component savings in mass production is obvious.

Suppose a more durable alternative architecture costs just €15 more per engine.

At 1.2 million engines, that is €18 million.

A tiny difference in unit cost becomes an extraordinary number when repeated across mass production.

The spreadsheet looks beautiful.

Now expand the calculation to include the entire life of the product.

Warranty claims, repairs, labour, replacement components, compensation programmes, engineering changes, customer dissatisfaction, damage to brand confidence and the commercial consequences of reduced trust all belong in the economic evaluation.

Some appear years after the vehicle has been sold.

Some affect future purchases rather than the vehicle that created the original problem.

The €18 million is not the value of the decision.

It is the value of one variable within it.

The spreadsheet may be right. The model behind it may be too small.

That is the real problem.

Arithmetic cannot repair an incomplete understanding of a system.

This is why price and cost are not interchangeable terms. A cheaper supplier may create higher total expenditure. A smaller maintenance team may make maintenance more expensive. A lower staffing budget may reduce operating capability. A software licence that costs less today may create a dependency that becomes extraordinarily expensive to remove.

And a component that costs more to purchase may be cheaper to operate over the required service life.

Possible, maximum and optimum are three different things.

The same applies to people, factories, infrastructure and entire enterprises.

A DEADLINE DOES NOT MAKE A SYSTEM READY

There is another variable that management frequently mistakes for an engineering capability: time.

Schedules are essential. Large projects would be impossible to coordinate without them. But a schedule is a model of when work is expected to be completed, not a mechanism capable of making that work complete.

I have repeatedly encountered the same absurd situation on industrial projects. A contractor explains that an activity requires six weeks using the currently available people, equipment and installation sequence.

Management replies that it must be completed in two.

Very impressive.

What exactly changed?

The number of workers did not increase. The available equipment did not multiply. The prerequisites remain incomplete and the day still contains twenty-four hours.

Only the deadline has changed.

If completion within two weeks is genuinely necessary, the project needs additional capability. That may require more employees, additional shifts, different equipment, subcontractors, alternative sequencing or a different construction method.

All of those options have costs.

Some may be economically justified, particularly when the alternative is a substantial delay to the entire project.

But the decision must change the system, not merely the date.

A deadline does not create capacity.

And there is a limit beyond which this becomes something more serious than inefficient project management.

A system must reach the required state of readiness before it is operated.

A deadline is not a readiness criterion.

This is particularly important in infrastructure and industrial installations, where readiness is determined by the state of interconnected physical and safety functions. A boiler, turbine, electrical installation, ventilation system, fire-protection system and industrial control system cannot simply be treated as independent items on a Gantt chart.

Their dependencies determine whether the plant can operate safely.

I remember the commissioning stages of a large power-plant project in which this distinction became painfully real. Some parts of the plant were being brought into operation while important surrounding systems were still incomplete.

The fire-protection contractor was struggling to complete its work. In my assessment, only a relatively small part of the overall delay was actually attributable to that company, yet it took responsibility for resolving the immediate problem.

I began helping its teams.

We reorganised the workforce to increase available working hours, and some employees voluntarily worked from seven in the morning until eight in the evening to accelerate completion.

Those people worked extraordinarily hard.

Their effort helped shorten the path to readiness.

It did not change the fact that the system was not yet ready.

At one stage, test operations proceeded without the completed fire-protection installation, with three fire-service crews deployed as a temporary measure. During subsequent commissioning work, the plant was operated in a building whose ventilation was still incomplete.

I had control over four roof-mounted fans, which we drove to their maximum available performance. We also opened the air intakes under our control.

Another twelve fans were connected to fire-protection functionality that had not yet been commissioned, so they were unavailable.

The temperature beneath the roof of the boiler house became extreme. Welders and other crews were still working inside, in conditions approaching 60°C.

Management wanted us to do something about it.

We did everything the available installation allowed.

We were not magicians. Four fans cannot automatically replace sixteen, just as a management instruction cannot create missing ventilation capacity.

The fundamental problem was not the effort of the people working on the installation. They were doing what they could, often far beyond ordinary expectations.

The problem was that operating conditions had been created before the surrounding systems could support them.

On another occasion, a boiler-related failure caused extensive flooding of the building, including electrical cabinets, automation and control equipment. The consequences propagated into installations that had not caused the original failure. Subcontractors were then confronted with questions about why their equipment had not been adequately protected.

This is exactly what happens when a project is treated as a collection of separate contractual responsibilities rather than one interconnected technical system.

A failure in one part of the installation does not respect the boundaries between subcontractors.

Water does not care who supplied the control cabinet.

Heat does not care which company was responsible for ventilation.

An unfinished fire-protection function does not become operational because somebody has found three fire engines.

In a separate industrial incident, a high-pressure pipe weld failed, releasing enough energy to damage walls within the building. Fortunately, nobody was killed.

But the absence of fatalities is not evidence that the situation was acceptable.

It means nobody was killed.

There is an enormous difference.

What is the purpose of accepting exposure to consequences like these merely to satisfy a date or demonstrate that another milestone has been completed?

A deadline does not create capacity, and it certainly does not create technical readiness. You can accelerate the work required to achieve readiness, but if the necessary functions and safety conditions are not available, you wait.

The schedule must follow the actual state of the system.

Not the other way around.

The absurdity becomes even clearer in less dangerous examples. Imagine a building whose walls are painted before its roof is complete. The painting contractor finishes the work and the milestone becomes green.

Then rain damages the walls and they need painting again.

The contractor may complete the same nominal activity more than once while the building remains unfinished.

The local task is complete.

The global project has made no equivalent progress.

A similar contradiction occurs when every contractor receives an individually plausible deadline but the dependencies between their tasks make the combined programme physically impossible.

HVAC cannot install equipment into a structure that does not exist. Heating cannot operate without its required medium. Commissioning cannot proceed because an upstream installation was scheduled to be finished yesterday.

The fact that a prerequisite was supposed to exist is not evidence that it exists.

This is not sophisticated mathematics.

It is the basic difference between a schedule and reality.

And the same mistake appears in industries where the consequences are measured primarily in money and reputation rather than physical damage.

WHAT DID THE RELEASE DATE OF CYBERPUNK 2077 ACTUALLY COST?

The launch of Cyberpunk 2077 is one of the clearest examples of a company achieving a planned milestone without achieving the required state of the product.

CD Projekt released the game on 10 December 2020. It was an enormous commercial event, selling 13.7 million copies by the end of that year.

The demand existed. The marketing had been effective. The product had already attracted extraordinary attention.

But the game was not ready.

The release was accompanied by serious technical and performance problems, particularly on older consoles. Refund arrangements followed, and Sony removed the digital version from the PlayStation Store on 18 December 2020.

CD Projekt then devoted substantial resources to patches, stability improvements, technical reconstruction and changes to gameplay systems.

The launch date had been achieved.

The required product state had not.

A large game is an exceptionally complex system. It contains interacting software components, data structures, assets, animations, artificial intelligence, rendering pipelines, physics, scripts, interfaces, platform-specific implementations and enormous quantities of content.

A small error can produce consequences somewhere completely unexpected. A change that resolves one problem can create another.

Anyone who has worked with sufficiently complex software understands how difficult it can be to determine why an apparently trivial modification has broken a completely different part of the application.

Estimating the precise completion date of such a project a year or two in advance is necessarily an estimate.

The number of dependencies is too large, and their behaviour cannot be predicted perfectly.

Yet companies continue treating an announced release date as if it were more important than the actual condition of the product.

CD Projekt should have delayed the release further.

That is my position.

A delay would have cost money. Development teams would have continued working, marketing would have needed adjustments and investors would have been frustrated. Customers would have complained, particularly after previous delays.

But customer frustration is manageable.

A broken relationship between a customer and a product is considerably harder to repair.

I would have preferred a different strategy: move the release date, explain the engineering reality, maintain the campaign and provide something worthwhile to the audience while the main product was being completed.

There were many possibilities.

CD Projekt already possessed one of the strongest entertainment properties in the industry through The Witcher. I can imagine a small, carefully designed piece of additional Witcher 3 content introducing an element of the Cyberpunk universe, perhaps allowing Geralt to encounter something completely out of place in his world.

A relatively modest creative project, released at the right moment, could have turned part of the frustration surrounding a delay into genuine excitement.

The important condition would still have been that the new release date led to a complete, properly tested AAA product.

Delaying a game and then releasing it in the same condition would simply add another failure.

Rockstar Games provides a useful contrast.

The wait for Grand Theft Auto VI has been extraordinary, and its release has been delayed. Yet anticipation remains enormous, promotional materials continue attracting attention and pre-orders are available ahead of the announced 19 November 2026 launch.

Rockstar has publicly connected the additional development time with the level of polish it intends to deliver.

Whether the final game meets that standard will become clear when it reaches players.

But the decision to move the date rather than force the product into an unfinished release is the correct one.

The cost of additional development must be calculated.

It does not justify releasing a system before it is ready.

Now consider what actually happened to CD Projekt.

The success of The Witcher 3, followed by Hearts of Stone and Blood and Wine, established the company as one of Europe's most valuable game developers. Around the release of The Witcher 3 in 2015, CD Projekt was valued at approximately half a billion euros.

By November 2020, just before the release of Cyberpunk 2077, the company was worth approximately €8.4 billion.

With a properly finished game, reaching €11–14 billion was possible.

Instead, management released a game that was not ready.

The launch was a disaster.

The company's market valuation collapsed. At its lowest point in the following years, its shares had lost approximately three-quarters of their value from the 2020 peak.

Years passed.

The developers repaired and improved Cyberpunk 2077. The game sold tens of millions of copies. Phantom Liberty demonstrated that the world CD Projekt had created still possessed enormous commercial potential. The company remained profitable.

And yet, in October 2026, almost six years after the release, CD Projekt was worth approximately €5.8 billion.

It had not even recovered its pre-launch market valuation.

Almost six years.

CD Projekt had spent years building a reputation that allowed investors to believe in its next major production.

Then management decided to release an unfinished game and damaged that confidence.

Repairing Cyberpunk was not the same as restoring the belief that the company's next major product would be delivered properly.

And that distinction has enormous economic consequences.

The company can continue earning money from its existing games. It can develop new products and build anticipation around them. But demonstrating that it can once again deliver an outstanding AAA

release is a different achievement from repairing a product that should not have reached the market in that condition.

The Witcher 4 will arrive with that history behind it.

CD Projekt must prove again what investors previously believed it could do.

Management made a bad decision. And the company has been paying for it ever since.

That is not merely a matter of reputation.

Market capitalisation is not an abstract number that becomes irrelevant because it does not represent cash sitting in the company's bank account.

It is the market value of the company's equity.

An enterprise's valuation affects the wealth of its shareholders and can materially influence its financial flexibility. Shares can be used as consideration in acquisitions, to raise new capital or as part of employee incentive arrangements. The value, liquidity and volatility of those shares also matter when investors use them as collateral.

A company with stronger equity valuation may be able to finance a transaction while issuing a smaller percentage of its ownership.

A company whose value has collapsed has fewer attractive options.

The effect propagates.

A problematic product release damages customer trust. Reduced trust influences expectations of future performance. Those expectations affect valuation. Valuation affects the economics of financing, acquisitions and ownership. And those changes influence the strategic options available to the business.

The company may find itself operating with fewer possibilities than it would have possessed after a successful launch.

This is why looking only at the cost of fixing Cyberpunk is economically inadequate.

The full cost of the decision extends beyond repairs, refunds and development expenditure.

It includes the productive capacity occupied by recovery work, the commercial opportunities that could not be pursued in the same way, and the loss of confidence in the organisation's ability to deliver its next major product.

Phantom Liberty was an excellent demonstration of what the Cyberpunk universe could offer.

But it was the only major expansion ultimately developed for Cyberpunk 2077.

A successful launch of a complete base game could have created a much stronger foundation for several years of additional content and commercial development.

Instead, a significant part of the company's early post-launch capacity had to be spent improving what had already been sold.

That is an opportunity cost.

And there is another form of capital involved.

Trust.

Before Cyberpunk, customers and investors had strong reasons to associate CD Projekt with the quality demonstrated by The Witcher 3. After Cyberpunk, that trust could no longer be treated as unconditional.

Trust is another form of operational capital.

It can take years to accumulate and one decision to consume.

A company may never receive an invoice labelled cost of lost trust.

That does not make the economic effect any less real.

So compare the complete system cost of a delayed, properly finished game with the cost of the premature release that actually occurred.

One option required additional development expenditure and changes to the commercial schedule.

The other produced years of repair work, consumed reputation, occupied future development capacity and damaged the market valuation of the entire enterprise.

CD Projekt chose the second.

So what did meeting that release deadline actually cost?

That question should have been answered before the game was released.

Not six years later.

THE ARCHITECTURE OF WORK MATTERS AS MUCH AS THE WORK ITSELF

There is another lesson hidden in the industrial examples, and it concerns the way organisations depend on people.

Organisations frequently mistake exceptional human capability for the capability of the system.

A highly competent employee can remember information that should be maintained by software, coordinate dependencies that should be built into the workflow, solve contradictions that should have been identified during planning and compensate for missing management functions.

The organisation continues operating, and its performance can look surprisingly good.

But part of its effective capacity exists only because one person is absorbing work that should have been distributed through the architecture.

High competence can function as credit for a badly designed system.

When that person eventually leaves, the organisation discovers that the capacity it believed it possessed was never embedded in its processes, tools or organisational structure.

The reverse is equally interesting.

A well-designed execution architecture can allow less experienced people to outperform a considerably more experienced team.

Consider industrial cable installation.

An experienced five-person electrical crew may receive a drawing and begin solving the entire task while performing it. They identify the route, interpret the documentation, plan the movement of heavy cables, determine how to negotiate bends and crossings, arrange materials and resolve unexpected problems.

Much of the crew's working time is consumed by decisions rather than installation.

Now compare that with a three-person crew whose work has been properly prepared beforehand.

Before the crew begins pulling cable, the routes are defined, cable bundles are prepared and measured, individual cables are grouped and labelled, and the material is staged for the next operation. Rollers are installed at difficult bends, and the team receives a clear sequence of work.

Instead of asking the installation crew to solve the entire problem repeatedly, much of the cognitive preparation has been moved upstream.

Under those conditions, a three-person crew with comparatively little experience can outperform a larger, experienced team.

That does not mean inexperienced electricians are superior to experienced electricians.

It means the architecture of the task can have a greater effect on the outcome than the experience difference between the people executing it.

The same principle applies to support functions.

A warehouseman, driver, office assistant or additional foreman appears as an immediate payroll expense. It is easy to remove that expense from a budget.

But if the absence of a warehouseman causes the most valuable technical person on the project to spend an hour searching for screws, the real system cost may be much higher.

If a project lead has to drive thirty kilometres to collect basic supplies, the absence of a driver or an effective logistics process has consumed capacity that could have been used elsewhere.

The question should not be limited to what a support employee costs.

The correct question is what the entire operation costs with that function and without it.

This is why headcount is not capability.

Adding employees does not automatically improve a company, and reducing employment does not automatically make it efficient.

The effect depends on what those people do, how the work is organised, what capacity they create and what other resources their presence or absence changes.

Sometimes the correct architectural decision is to hire more people. Sometimes it is to reduce headcount. Sometimes it is to train existing employees, redesign responsibilities or replace an individual who cannot perform the required work.

And sometimes the problem sits much higher in the organisational structure.

A manager can become a bottleneck. They can delay decisions, suppress information, reject valid technical observations, assign responsibilities incorrectly or create an environment in which competent employees stop reporting problems because experience has taught them that nobody will listen.

Consider an operator who repeatedly notices a machine behaving differently when processing a particular material.

The observation travels upwards.

The manager dismisses it without investigation.

After several unsuccessful attempts to communicate the issue, the operator stops reporting it.

The organisation has not eliminated the problem.

It has eliminated part of its ability to see the problem.

Replacing that manager may create more value than purchasing another piece of equipment.

In another company, the manager may be excellent and the actual problem may be the workload, the process or an unrealistic operating requirement.

Without analysing the system, we do not know which.

That same reasoning applies to software.

A vendor may propose a product that promises increased productivity, better management or higher efficiency. The product may be technically excellent, but that does not establish that the company needs it.

An application changes specific functions. Those functions interact with other parts of the enterprise. A new platform may remove manual work, but it may also introduce new dependencies, new maintenance requirements, different decision paths and additional complexity.

The proper starting point is therefore not choosing the software.

It is identifying the function the system is missing.

Sometimes the required solution is a new application. Sometimes it is a small integration between existing tools. Sometimes the organisation already possesses the required information but fails to make it available where decisions are actually made.

A vendor may offer a €500,000 system when the company needs only a few specific functions that can be implemented within its existing environment at a fraction of the cost.

But that can only be established after understanding the architecture.

Buying software before identifying the required function is simply choosing the tool before understanding the work.

And this brings us to another mistake frequently made when analysing large organisations.

Experience with one facility is not knowledge of every facility.

Imagine a company operating ten factories. They use similar machinery, manufacture similar products and belong to the same corporate structure.

Are they ten identical systems?

Of course not.

Their physical layouts differ. Their equipment has accumulated different operating histories. Their personnel have different skills and experience. Workstations have been modified, maintenance practices have evolved and individual managers have developed different ways of organising work.

Even the same machine can behave differently after years of operation under different conditions.

You analyse the first factory and learn an enormous amount about the company. That knowledge makes the second analysis faster.

By the fifth or sixth facility, much of the technology, documentation and organisational context is already familiar. A competent team can collect, organise and examine substantial information before the architect arrives.

The first facility may require a month.

Another may require ten days.

A later factory may require only three because most of the preparatory work has already been completed.

But I still need to see the factory.

I need to understand how that particular facility actually operates, how its people perform their work and how its physical and organisational dependencies behave.

Experience makes the next analysis faster.

It does not make the next analysis unnecessary.

Having worked in an aircraft factory does not mean I understand every aircraft factory.

It means I understand the facility I examined, its technology and its particular operating conditions. That experience gives me a substantial advantage when entering another facility.

It does not give me knowledge of a system I have not yet seen.

Even ten factories belonging to the same company remain ten individual operating systems within a larger enterprise.

Optimising the group requires understanding both their individual architectures and the dependencies between them.

SECTION 09

ENTERPRISE / SYSTEM ARCHITECTURE

A GOOD METHOD IS NOT A UNIVERSAL RESULT

This brings us back to consultants, software vendors and the extraordinary confidence with which improvement percentages are advertised.

A methodology may be perfectly legitimate. A software product may perform its intended function exceptionally well. A machine manufacturer may offer a genuine technical advance.

But the existence of a good solution does not establish that it is the right solution for your company.

Suppose a manufacturer offers ceramic rollers that reduce wear and maintenance requirements.

Their equipment works exactly as advertised.

Installing it across a particular production system requires €300,000 in capital expenditure and produces €15,000 in additional recurring annual financial benefit.

The simple payback period is twenty years.

The rollers are excellent.

The investment is not.

At least, not on those economics.

Another factory may obtain a much larger benefit because its equipment operates under different conditions. A third may achieve little direct financial improvement but substantially increase the reliability of a critical process.

The same product.

Different systems.

Different consequences.

The same applies to software.

A system reduces the time needed to complete a particular administrative task by 50 per cent.

Excellent.

We have identified a measurable local improvement.

What happens to the released time? Does the company require fewer working hours? Does the employee perform additional useful work? Does the task simply stop being the bottleneck while the rest of the process remains unchanged?

Does the new software introduce recurring costs or dependencies elsewhere?

The software's advertised local performance does not establish its enterprise-wide economic effect.

And it certainly does not establish a 20 per cent improvement in company profitability.

This is the fundamental weakness of universal business recipes.

A method proven in one environment is not automatically optimal in another. The fact that it has worked before establishes that it works under certain conditions. It does not establish the complete consequences of applying it to a different system.

Knowledge of an industry is useful. Knowledge of its technologies, common operating models and typical problems is valuable. Broad cross-industry experience can make it easier to recognise transferable functions and identify possibilities that specialists working within one professional category might overlook.

But none of that eliminates the need to understand the particular company.

I can understand how a technology works without knowing whether it is appropriate for your business. I can recognise a defective process from publicly available information without knowing the financial effect of repairing it. I can identify an opportunity to improve operations without claiming to have discovered the global optimum of an enterprise I have not reconstructed.

These are not contradictions.

They are different levels of knowledge.

The ability to calculate a local improvement is not the same as the ability to predict its consequences across an enterprise.

And understanding those consequences is not yet the same as establishing the optimum under the company's actual constraints.

The precision of an answer cannot exceed the precision of the model behind it.

If the model supports a reliable conclusion, give the conclusion.

If it supports a numerical estimate, calculate it.

Account for recurring operating costs, test the assumptions and use an appropriate margin of safety. If a realistic upside appears to be five per cent but only two per cent can be established with sufficient confidence, present the conservative figure rather than building a business case on everything going perfectly.

Then evaluate the required capital expenditure separately.

But if the necessary information is missing, do not invent the result.

It is perfectly acceptable to say that an intervention will improve a particular process while being unable to predict its complete financial consequences.

It is not acceptable to disguise that uncertainty with an impressive percentage.

OPTIMISATION DOES NOT MEAN MAXIMUM PROFIT

There is another assumption that deserves to be challenged.

When people hear the word optimisation, they often think of cutting costs, increasing output or maximising profit.

Those can be legitimate objectives.

They are not a universal definition of optimum.

A company can reduce its annual operating expenditure and become less resilient. It can increase machine utilisation and accelerate equipment deterioration. It can reduce headcount and lose essential operating capacity. It can increase short-term profitability while compromising its ability to function over the following decade.

A higher immediate financial result is not necessarily a better system.

Consider two possible architectures. One produces an annual operating cost of €820 million. The other costs €840 million but delivers substantially stronger operational resilience, better reliability and greater flexibility.

Is the second architecture worse because it costs €20 million more annually?

Not when that additional capability is necessary to meet the requirements of the enterprise.

If the company must maintain service through disruptions, protect critical operating functions, retain sufficient reserve capacity or avoid large future losses, the more expensive architecture may be the correct solution.

The requirements define the problem.

The system must be designed to satisfy them.

If a production line must sustainably deliver 2,000 units per hour, that requirement is not an invitation to debate whether 1,500 would be more convenient.

It is a constraint to be incorporated into the design.

The task is to determine what machinery, personnel, maintenance capacity, quality control, logistics, information flow and operating reserves are necessary to achieve the required output.

If the requirement changes, the architecture may change.

If the acceptable risk changes, the architecture may change.

If the available capital changes, the architecture may change.

An enterprise designed for sustainable performance over fifteen years may require different decisions from one designed to meet a short-term production constraint.

The objective is not to maximise every parameter.

It is to find the combination that produces the best system-level result under the actual constraints.

This is why the lowest annual operating expenditure does not necessarily represent the optimum. Resilience is not an optional luxury added after optimisation.

It can be part of the optimum itself.

Nor does every visible improvement constitute optimisation.

If an industrial staircase lacks a handrail, you can recognise and correct that problem without reconstructing the entire enterprise. If a production process contains an obvious unnecessary movement, it may be possible to eliminate it immediately. If a machine is operating with a defective component, replacing that component may be an entirely appropriate local improvement.

None of those actions establishes that the entire company has been optimised.

Improvement and optimisation are different claims.

A local improvement requires sufficient understanding of the affected process and its relevant dependencies.

A claim about the optimum of an entire enterprise requires a correspondingly broader model.

This also explains why optimising a company is not synonymous with increasing its profitability.

Sometimes the most important improvement is preventing a business from failing.

A company can be stabilised without becoming exceptionally profitable. Restoring its ability to operate, maintain production, meet obligations and survive future disruption is a valid architectural objective in its own right.

Confusing survival, profitability, growth and optimisation produces bad decisions because these are different objectives with different consequences.

A system cannot be declared optimal before its purpose and constraints have been established.

WHY DO WE DESIGN BUILDINGS BUT NOT COMPANIES?

There is one final aspect of enterprise architecture that deserves attention.

Most of this article has concerned companies that already exist.

Their processes, technology, buildings, personnel and dependencies have developed over time. Some elements were designed deliberately. Others were added to solve immediate problems.

Temporary arrangements became permanent. New technology was attached to old technology. Departments grew, responsibilities changed and the organisation accumulated an operating structure that nobody necessarily designed as a coherent whole.

Before such a company can be optimised, its actual architecture has to be reconstructed.

But what about a company that does not yet exist?

When someone decides to construct a substantial building, they employ an architect.

The owner may know exactly what the building should accomplish. They may understand its purpose, its commercial value, its intended users and the available budget.

That does not mean they can design its complete physical architecture.

The architect does not replace the owner's knowledge of the intended use. The architect translates that knowledge into a coherent design within the physical, technical and economic constraints of the project.

Yet people routinely invest substantial capital in new businesses without commissioning an equivalent examination of how the enterprise itself should function.

They employ financial specialists, technology specialists, process engineers, logistics experts and managers.

Each may be highly competent within a particular area.

But who is responsible for designing the relationships between all those areas?

A company can employ excellent specialists and still have a poor overall architecture.

Not because those people are incompetent, but because excellence in individual components does not automatically create excellence in their combined operation.

And unlike an existing enterprise, a new business does not necessarily require the reconstruction of decades of accumulated mistakes.

Its architecture can be designed before the initial investment commits the organisation to particular machinery, processes, buildings, staffing structures and dependencies.

Of course, the appropriate scope depends on the investment.

A small business established with €100,000 does not need the architecture of an industrial corporation investing hundreds of millions. Its planning horizon, complexity, operating model and capital limitations are completely different.

The correct architecture is not the largest, most expensive or most technologically sophisticated design imaginable.

It is the best design for the actual mission and the available resources.

This is another reason not to confuse architecture with replacing management's expertise.

The owner may understand the market exceptionally well. The production director may know the technology better than anyone else. The financial director may understand the company's capital structure in extraordinary detail.

Architectural analysis does not make those competencies redundant.

A SYSTEM-LEVEL QUESTION

Where does the decision really create value?

It integrates them into a larger system.

And the larger the company becomes, the more expensive it can be to discover that nobody designed the relationships between its parts.

IT DEPENDS

We can now return to the original question.

How do you optimise a company?

The first requirement is to reconstruct the system that actually exists.

That means understanding the organisation beyond its formal documentation. We need to know how information moves, how decisions are made, where people compensate for missing functions, what equipment is capable of doing, how workflows interact, where costs arise and how changes propagate between different parts of the enterprise.

A dependency graph can become an extremely valuable representation of that reality. It allows proposed changes to be placed into an existing structure. A new machine, supplier, product or software function can be analysed through the dependencies it changes.

But the graph is a map.

It does not replace understanding.

A relationship between two components tells us that they interact. We still need to know what the interaction means, how it behaves and under what conditions it matters.

Only after reconstructing the relevant reality can we establish the constraints and determine the desired operating state.

Those constraints may be physical, commercial, financial, regulatory or strategic.

The objective is not to maximise every parameter. It is to find the combination that produces the best system-level result under the actual constraints.

That process may lead to substantially lower expenditure.

It may also lead to higher expenditure, more employees, more expensive machinery, additional redundancy, greater spare capacity or a completely different production strategy.

None of those outcomes is inherently good or bad.

They are possible answers to a specific system problem.

The important thing is that the answer emerges from understanding the enterprise, rather than being imposed upon it.

And that is where the entire argument comes together.

A company can increase production without becoming more efficient. It can reduce expenditure without saving money. It can achieve every scheduled milestone without making genuine progress.

It can prepare itself for sale while becoming a worse business.

It can deliver a product on the announced date while destroying part of the value that product was supposed to create.

It can employ exceptional people and mistake their ability to compensate for a broken organisation for evidence that the organisation itself is well designed.

And it can spend millions on sophisticated software without acquiring the functions it actually needs.

The common error is always the same: treating a visible local result as proof of a successful system-level decision.

Every effect has a cost.

The full cost includes the resources necessary to create it, the architecture required to sustain it, the consequences it produces over time and the opportunities it may prevent or enable.

Until those variables are sufficiently understood, the financial result remains incomplete.

That is why a beautiful dashboard cannot establish that a company is well optimised, a lower invoice cannot establish that money has been saved, and a completed milestone cannot establish that the system is ready.

There is no point demanding a deterministic answer from an incomplete model of reality.

So when somebody asks me how I would optimise their company, I will not begin with a recommendation to cut costs, increase output, reduce employment or install another platform.

I will begin by finding out what company actually exists.

Then we can establish what it should become.

There is no universal recipe for optimising a company. There is only a disciplined way of finding the right answer for a particular system.

And until that work has been done, the answer to the original question remains exactly the same.

THE ANSWER TO THE ORIGINAL QUESTION



IT DEPENDS.

ARCHITECTURE FIRST. NUMBERS AFTER.

Chrzanowski

