

# MOST COMPANIES DO NOT OPTIMIZE THE COMPANY. THEY OPTIMIZE THE SPREADSHEET.

**A better KPI is not proof of a better company.**

Consulting can improve the report.

**Architecture improves the company.**



**MATEUSZ CHRZANOWSKI**

SYSTEM OF SYSTEMS ARCHITECT

STRUCTURE / FLOW / RESPONSIBILITY / CONTROL



# REPORTING IS NOT OPTIMIZATION

Most companies do not optimize the company.

They optimize the spreadsheet.

A consulting firm enters, charges tens or hundreds of thousands of euros, prepares a beautiful presentation, redesigns a few KPIs, creates new processes, adds dashboards, runs workshops, identifies “quick wins” and gives the board a feeling of control.

The report looks better.

The chart starts moving in the right direction.

The spreadsheet becomes cleaner.

But a better KPI is not proof of a better company.

You can pressure people and improve numbers. You can cut costs and improve numbers. You can tighten reporting and improve numbers. You can force a system to produce better-looking metrics for a while.

The problem is that the hidden cost does not appear tomorrow.

It does not appear next week.

It compounds quietly: in turnover, indifference, cynicism, hidden errors, passive sabotage, lower loyalty, lower quality, less responsibility and thousands of small leaks nobody sees in Excel.

A KPI can improve while the company becomes weaker.

That is the difference between spreadsheet optimization and architecture-led company redesign.

Consulting can improve the report.

Architecture improves the company.

## SPREADSHEET OPTIMIZATION

### VISIBLE KPI

Report improves.

Hidden system cost may rise.

## ARCHITECTURE-LED REDESIGN

### REAL CAPABILITY

People, tools, flow and control improve together.

COMPANY ARCHITECTURE / OPERATING SYSTEMS

# PROFIT IS NOT ARCHITECTURE

---

A company can be profitable, growing and proud of its results while still operating on a broken architecture — or no architecture at all.

Profit is not proof of good architecture.

Sometimes profit only proves that the market still tolerates the waste hidden inside the system.

A company may have a strong product, a good brand, loyal clients, a favourable market moment or people who compensate every day for problems the system should never produce. Revenue can hide structural weakness for years.

A STRONG RESULT CAN HIDE

## STRUCTURAL WEAKNESS

- GOOD PRODUCT
- STRONG BRAND
- LOYAL CLIENTS
- FAVOURABLE MARKET
- PEOPLE COMPENSATING

PROFIT ≠ PROOF OF GOOD ARCHITECTURE

# STRUCTURE OR SEDIMENT

Most companies are not designed as systems.

They accumulate.

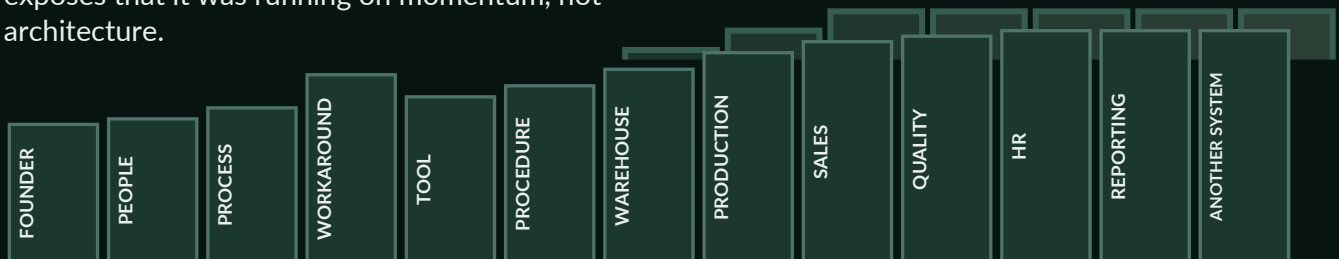
First there is a founder. Then a few people. Then a few processes. Then a workaround. Then a tool. Then a procedure. Then a warehouse. Then production. Then sales. Then quality. Then HR. Then reporting. Then another system added on top of the previous one.

After years, the board looks at the company and sees structure.

An architect sees sediment.

Layers that were never designed together. Flows that grew by accident. Software that forced the company into its own limitations. Procedures that exist on paper but have been replaced by informal reality. People keeping the company alive despite the system, not because of it.

A company can look healthy until the first real shock exposes that it was running on momentum, not architecture.



**THE BOARD SEES STRUCTURE.**

**THE ARCHITECT SEES SEDIMENT.**

# THE COMPANY IS A MULTILAYER SYSTEM

A company is not a spreadsheet.

It is not an org chart.

It is not a set of procedures.

It is not software.

It is not a production line.

It is not culture posters.

A company is a multilayer operating system: people, roles, decisions, procedures, tools, machines, software, physical space, logistics, quality, information flow, reporting, finance, risk, loyalty, communication, responsibility and crisis resistance.

The biggest losses often do not happen inside one layer.

They happen between layers.

Between production and warehouse. Between warehouse and purchasing. Between machine and report. Between foreman and manager. Between procedure and reality. Between person and tool. Between shop floor and headquarters. Between board decision and what actually happens at the bottom.

A company loses value between layers.

That is where architecture has to work.

A systems architect is not there to write in a report that "Department A does not communicate well with Department B."

That is symptom-level thinking.

The architect redesigns the structure so the information does not have to travel through a weak channel in the first place. The architect defines the flow, the tool, the responsibility, the status, the decision point, the escalation path, the report and the system response.

My work is not to describe friction.

My work is to redesign the structure that produces it.



VALUE IS LOST AT THE INTERFACES  
ARCHITECTURE WORKS BETWEEN LAYERS

# ANALYSIS STARTS AT THE BOTTOM

Real analysis does not start in the boardroom.

It starts at the bottom.

With the gatekeeper. The operator. The warehouse worker. The driver. The foreman. The production worker. The person closest to real work, real error, real delay, real waste, real workaround and real quality.

Only then should the analysis move upward: team leaders, shift managers, department heads, planning, quality, logistics, sales, administration, finance and finally the board.

The board sees the company as a report.

The lowest layers see the company as reality.

Very often, the people at the bottom know exactly how the company really works. They know which procedure is dead. Which tool creates friction. Where material waits. Who bypasses the system every day. Which report is fiction. Where the machine really loses time. Where a manager formally controls something but operationally sees nothing.

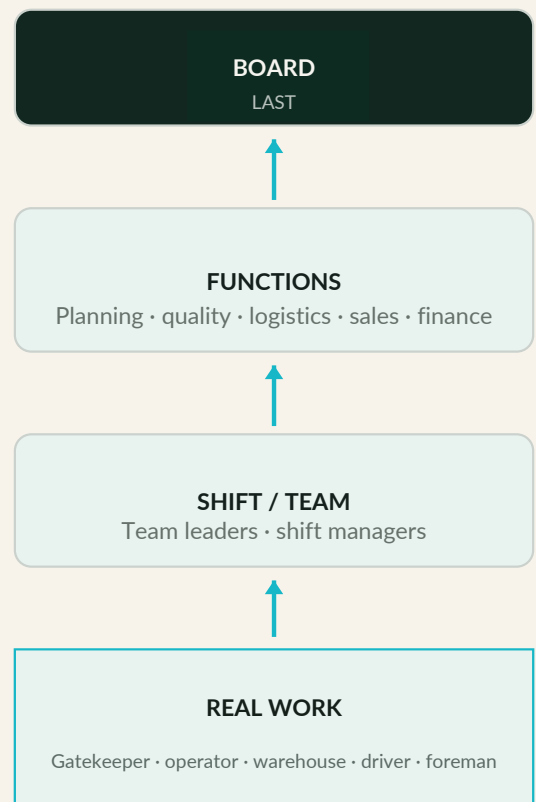
A company is not what the board sees in a presentation.

A company is what people at the bottom must do so the system somehow keeps working.

That is why mapping must move from the bottom up.

Otherwise, you create another presentation about the company, not a picture of the company.

ANALYSIS DIRECTION: BOTTOM → UP



# A TOOL IS NOT ARCHITECTURE

One of the biggest mistakes in optimization is confusing a tool with architecture.

A new procedure is not architecture.

A new software system is not architecture.

A dashboard is not architecture.

An app, an AI tool, a new ERP, a new report, a new department structure — all of these can become elements of architecture, but none of them create architecture by themselves.

A procedure is only an instruction inside architecture.

A good procedure is not paperwork.

It is an interface between the worker and the system.

It must be clear enough to be understood by the person who has to execute it under pressure: on a production line, in noise, in stress, in a truck, on a construction site, in a warehouse, next to a machine or in front of a client.

Software must not become a god.

A company should not kneel before the limitations of a program.

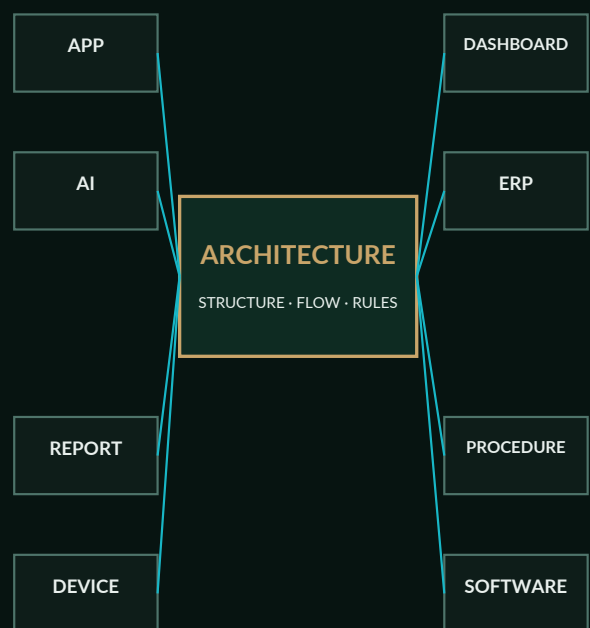
Software should execute the architecture of work.

When the right software does not exist, it should be designed. When the right device does not exist, it should be designed. When an existing tool creates friction, the tool should be changed. When technology from another industry solves the problem, it should be moved into the system where it makes sense.

I do not work on hype.

I do not care what is fashionable.

I care what works according to the logic of the system.



TOOLS EXECUTE FUNCTIONS.

ARCHITECTURE DEFINES THE SYSTEM.

# EXECUTION ARCHITECTURE

A good example is an industrial construction site.

On a large industrial site, classical crew management becomes operationally dead very quickly.

Even a housing estate construction site is difficult: multiple buildings, raw concrete, poor markings, changing access routes, staircases, materials, deliveries, crews and constant collisions. An industrial construction site is another class of complexity.

In one of my strongest real-world execution cases, the site included 21 buildings, external installations, roof installations, work at heights of 30–45 meters, difficult-access routes, a huge spread of people and a technical scope that many people found unreadable.

A full control walk through the site could take around six hours, often even with a car.

On that kind of site, information cannot walk around the site together with a human being.

Information has to move faster than the human.

So the lowest operational unit had to be designed differently.

**21**

BUILDINGS

**30–45 m**

WORK AT HEIGHT

**~6 h**

CONTROL WALK

**INFORMATION HAS TO MOVE  
FASTER THAN THE HUMAN.**

# THE FIELD EXECUTION CELL

A brigade was not “a group of people somewhere on site.” It was a small field execution cell: foreman plus two workers. The foreman was not standing with a clipboard. He was working with the crew, which meant he had to execute work, receive instructions, report issues, send photos, receive corrections and lead the team at the same time.

That required information architecture and tool architecture.

Radios. Network-connected tablets. Active documentation. Active reports. GPS and work-time registration. Live issue reporting. Field photos. Corrections drawn directly on images. Immediate instructions pushed back to the brigade.

A tablet without architecture is just a screen in a rubber case.

A tablet inside active documentation, reporting, communication, GPS, responsibility and live correction becomes a field command terminal.

This mattered because the people were not experienced industrial automation workers. There were miners, drivers, a gardener and people without construction, electrical or automation background. Yet they were executing a technically demanding AKPiA / HVAC scope on critical infrastructure.

Industrial automation is not laying ceramic tiles.

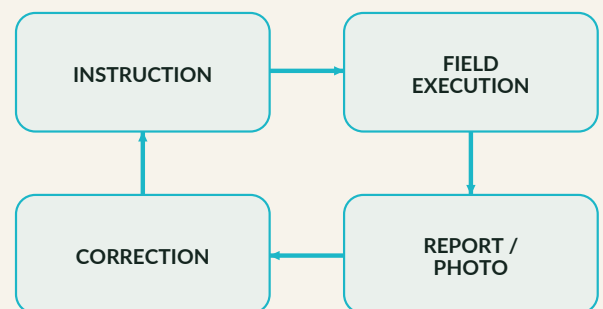
The work had to be decomposed into execution logic: what to do, in what order, with which tool, where, how to check it, how to report it, how to correct it and how to move into the next layer.

That was not task management.

That was execution architecture.



FOREMAN + 2 WORKERS



RADIO • TABLET • ACTIVE DOCUMENTATION  
GPS • LIVE REPORTING • LIVE CORRECTION

# MOBILE LOGISTICS NODE

At a certain scale, communication is not enough.  
Logistics becomes architecture.

If a brigade has to walk to the warehouse, find a tool, collect material, return to the object, reach the working area and only then continue the task, the system loses time, energy and rhythm.

A warehouseman with a car, available on radio, is not “someone who drives things around.”

He is a mobile logistics node.

He delivers tools. He delivers material. He brings ladders. He moves people. He supports brigades in the field. He reduces dead walking. He limits chaos around warehouses. He protects access to resources. He prevents work from stopping just because someone has to cross half the site for one missing element.

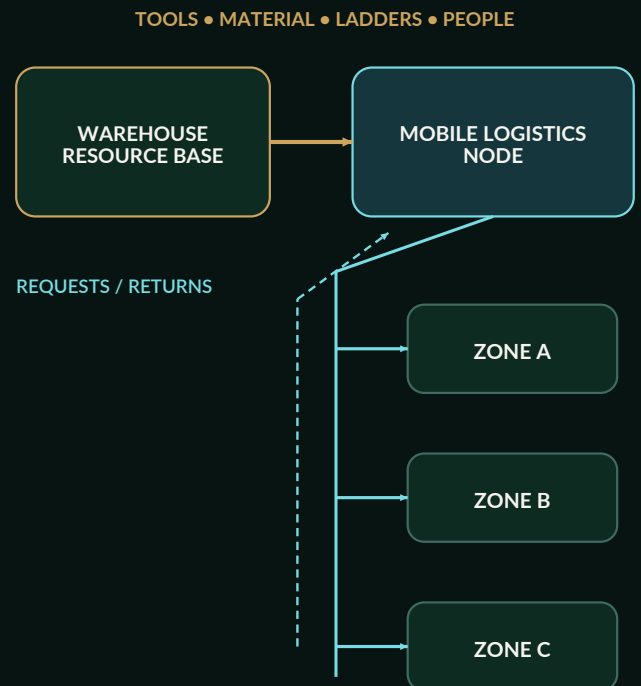
Access to resources has to be designed.

It cannot be wild.

The same applies to tools. Special tool backpacks for brigades may look expensive until you understand the operating environment. On a large, multi-level industrial site, they are an efficiency layer. A brigade has to be mobile and tool-independent. If an expensive backpack eliminates walking, waiting, searching and work interruption, it is cheaper than chaos.

The purchase price is not the real system cost.

The real system cost is the effect of the tool on work flow.



LESS WALKING • LESS WAITING

LESS SEARCHING • LESS INTERRUPTION

MORE EXECUTION TIME

# HARDWARE IS ALSO ARCHITECTURE

Hardware is also architecture.

On paper, many tablets look similar. In an industrial environment, the difference between a reliable device and a failing device is huge.

Some tablets more expensive than a basic iPad failed badly in that environment. They froze, slowed down, created frustration and became operational friction. And friction in that system was not a small irritation. It was an execution loss.

Today, for that class of industrial site, I would standardize the field-terminal layer on basic iPads in rugged 360-degree cases, with shoulder-carry capability.

Not because the logo is fashionable.

Because hardware reliability is part of execution architecture.

If a tool is part of the system, it must fit the environment, the people, the rhythm of work, the risk of falling, dust, rain, gloves, time pressure, documentation and real use.

A badly selected tool does not support architecture.

It becomes another source of loss.

## FIELD HARDWARE FIT

### RELIABILITY

Does it keep working?

### ENVIRONMENT

Dust • rain • gloves • falls

### RHYTHM

Does it slow execution?

### PEOPLE

Can the real user operate it?

**BAD TOOL → EXECUTION LOSS**

# SOFTWARE EXECUTES THE ARCHITECTURE

The same logic applies to software.

In my model, the IT team should not guess the product. They should not invent the business logic, dependencies, states, user paths, permissions, system reactions, reports and data flows on their own.

The IT team should code the architecture.

I design software from the side of use logic, system role and logical interface. Not graphic UI first. Logical interface first: what the user sees, when, why, after which action, with which status, and what that means for the next user, process, report and decision.

My graphs are dense, but readable. They show dependencies, flows, screens, reactions, events, transitions and system logic before code.

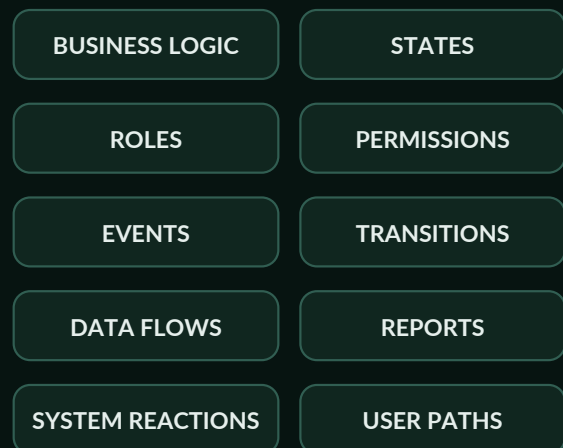
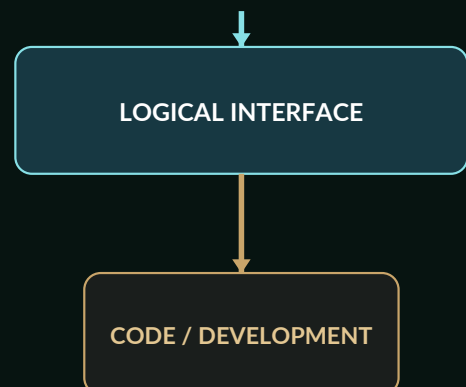
That is why a large system can be built with a small team: one mid-level developer and three juniors. Not because the product is simple. Because the architectural burden is not pushed onto developers.

Developers do not receive fog.

They receive structure.

This is not a mockup.

This is system logic before code.



**IT CODES THE ARCHITECTURE.**

# PHYSICS IS THE BOUNDARY

Systems architecture does not stop at processes.

Sometimes you need to design new software.

Sometimes a new device. Sometimes a new class of devices. Sometimes a factory layout. Sometimes production flow. Sometimes a quality standard.

Sometimes logistics. Sometimes a use case for AI.

Sometimes a human-robot work model. Sometimes the simplest possible system based on paper, a phone and a free tool.

There are two boundaries: physics and technology availability.

I do not care which industry a technology comes from.

I care what it makes possible.

A solution from aviation can work in production. A solution from heavy industry can work in a small workshop. A technology from one sector can create advantage in another if it fits the problem at the first-principles level.

Industry is a label.

Physics is the boundary.

That is why hype does not interest me.

AI makes sense where it emerges from the system architecture. Humanoids make sense where they genuinely increase flow, safety or productivity. Automation makes sense where it removes a real structural problem, not where it looks good in a presentation.

Sometimes the right answer is AI.

Sometimes it is an iPad.

Sometimes it is a warehouseman with a car.

Sometimes it is a €350 seat cover.

Architecture decides.

**AI**

when it fits

**iPAD**

when reliability fits

**MOBILE NODE**

when logistics fits

**€350 SEAT COVER**

when simple works

## TWO BOUNDARIES

PHYSICS

TECHNOLOGY AVAILABILITY

**ARCHITECTURE DECIDES.**

# EFFICIENCY WITHOUT RESILIENCE IS FRAGILITY

Real optimization is not only about making a company faster or cheaper.

It is about making it harder to break.

Efficiency without resilience is optimized fragility.

A good company architecture does two things at the same time: it minimizes the risk of internal crisis and increases resistance to external crisis.

Internal crisis is usually the result of bad architecture: lack of visibility, wrong roles, wrong decisions, hidden dependencies, single points of failure, overloaded key people, weak logistics, dead procedures, invisible losses or poor communication between layers.

A well-designed company should not produce a crisis from the inside.

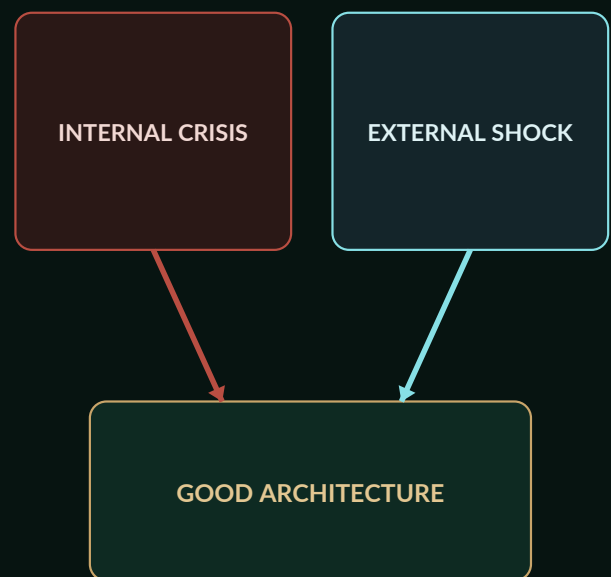
External crisis is different. There is no 100% guarantee against the market, war, energy shock, regulation, technology, financing pressure, demand collapse or supply-chain disruption.

But good architecture allows the company to absorb shock.

The company can change operating mode. Move resources. Maintain critical flows. Detect deviations faster. Reduce panic. Protect quality. Use people, data, tools and structure before a crisis turns into collapse.

Internal crises are usually architecture failures.

External crises are stress tests.



- DETECT
- ABSORB
- REALLOCATE
- PROTECT CRITICAL FLOWS
- CHANGE OPERATING MODE

# STRATEGIC MOAT

Good architecture also creates strategic moat.

A real advantage is not only that competitors cannot copy you.

A real advantage is that even a new technology does not make migration immediately rational.

This is especially visible in heavy, logistics-heavy and capital-intensive industries. A technology may become technically possible long before it becomes economically, operationally and structurally obvious for the whole industry.

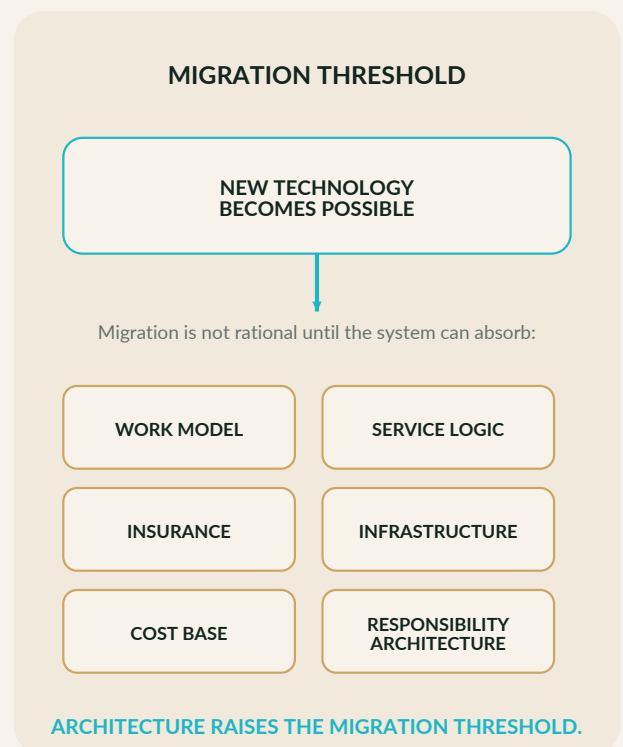
An autonomous truck may one day exist as a product. That does not mean the transport industry will immediately rebuild its work models, service logic, supervision layers, insurance structures, cost base, infrastructure and responsibility architecture.

If the existing operating model is already highly optimized, the new technology does not automatically create enough economic advantage to make migration obvious.

That is how architecture protects a company not only from today's chaos, but also from tomorrow's displacement.

A real moat is not only that competitors cannot copy you.

A real moat is that even a new technology does not make migration immediately rational.



# THE HUMAN LAYER

The most underestimated layer of company architecture is the human being.

Not HR.

Not employer branding.

Not value posters.

Not workshops about culture.

The human being as a system element: a source of quality, risk, loyalty, knowledge, initiative, early signals, responsibility and resilience.

A large company must be architected to see the small worker.

The board will not see the gatekeeper, the operator, the driver, the warehouse worker, the foreman or the person on the production line. The board will see reports. The human is seen by the middle structure: team leaders, foremen, managers, local HR, employee support, communication channels and properly designed response procedures.

A manager is not only a reporting layer.

A manager is the company's interface toward the human being.

A manager is also part of the company's sensory system.

They must know their people. They must notice who normally works well and suddenly becomes silent. Who has a private crisis. Who carries the team. Who poisons the team. Who needs help. Who deserves recognition. Who should be removed from the system.

A company that does not see people loses information before it becomes a report.

Then it acts surprised when a crisis appears "suddenly."



# VISIBLE COST. HIDDEN COST.

Employee support costs money.

A good gift costs money. A loan costs money. A support department costs money. Better truck equipment costs money. A company car for a developer costs money. Seat covers for an Actros cost money. A microwave, inverter, additional battery, good mats and a television in a truck cabin cost money.

But the cost of seeing people is visible.

The cost of not seeing people is hidden.

It appears in turnover, indifference, taking things from work, damaged equipment, hidden errors, silence, cynicism, passive sabotage, poor quality, low responsibility and the sentence: "it is just a company."

A company that treats people like replaceable parts should not be surprised when people treat the company like an anonymous wallet.

A large company that can see a small worker sends a message with enormous force:

We see you.

We appreciate you.

It is good that you are here.

That is not softness.

That is operational intelligence.

Human visibility is not charity.

It is operating architecture.

Employee loyalty is not a soft HR topic.

It is crisis infrastructure.

## VISIBLE COST

GIFT

LOAN

SUPPORT

BETTER EQUIPMENT

TIME

## HIDDEN COST

TURNOVER

INDIFFERENCE

ERRORS

CYNICISM

SABOTAGE

LOW QUALITY

LOW RESPONSIBILITY

**THE COST OF SEEING PEOPLE IS VISIBLE.  
THE COST OF NOT SEEING THEM  
IS HIDDEN.**

# LOYALTY IS CRISIS INFRASTRUCTURE

A company with 50,000 employees and 40,000 people genuinely loyal to the organization has a resource that cannot be bought on the day of crisis. Those people give the company time, calm, effort, knowledge, initiative, responsibility and willingness to go through a difficult period together with the organization.

Loyal people can keep a company alive even in a large crisis.

Not because they are naive.

Not because the company has nice slogans.

Because over years the company built a relationship in which the person feels that this is also their system.

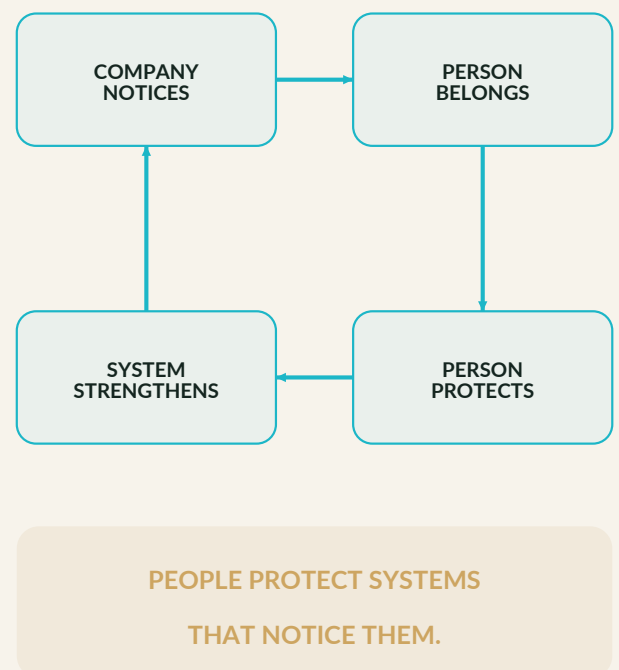
People protect systems that notice them.

In a good architecture, people start protecting the company from inside. They see when someone damages tools, hides mistakes, generates loss, poisons the team or acts against the organization. They report it because they no longer think: "that is their problem."

They think: "that hits us."

That is self-regulation.

It takes a long time to build. But when it starts working, it changes the company more deeply than another dashboard.



COMPANY ARCHITECTURE / OPERATING SYSTEMS

# GENEROUS TO CONTRIBUTION. RUTHLESS TO PATHOLOGY.

Human architecture does not mean indulgence.

A strong company is generous toward contribution and ruthless toward pathology.

If someone works well, helps others, protects quality, cares for tools, stays loyal, reports problems and genuinely brings value into the system, the company should see that person, appreciate them and strengthen them.

If someone breaks the rules, damages the system, cheats, steals, sabotages, poisons the team or acts against the company, they should leave the system.

There is no contradiction here.

Hardness toward pathology protects people who work well.

Recognition should not be turned into a contest. Contests create rivals. Real recognition creates belonging.

A good phone, good headphones, real help, a meaningful bonus, support during crisis, solving a small problem that destroys daily comfort – all of that can be part of loyalty architecture when done intelligently.

A gift is not valuable because it has a logo.

It is valuable because it proves that the company noticed a person.

## CONTRIBUTION

QUALITY

HELP OTHERS

CARE FOR TOOLS

LOYALTY

REPORT PROBLEMS

REAL VALUE

## PATHOLOGY

BREAK RULES

DAMAGE

CHEAT

STEAL

SABOTAGE

POISON TEAM

**STRENGTHEN.**

**REMOVE.**

# TRUST CHANGES TRANSFORMATION

A company that has built trust can also implement difficult change differently.

This includes AI, automation and workforce reduction.

AI transformation without trust becomes fear management.

If people see the company as a foreign body, every AI implementation will feel like a threat. Every automation effort will sound like: "they are coming for us." Every restructuring will become a war.

But a company that has been honest for years, seen people, recognized contribution and communicated seriously has a different position.

Even a large workforce reduction can be handled architecturally: communicated early, placed in context, supported by transition time, reskilling, retraining, job-search support and a departure structure that allows people to remember the company as an organization that treated them decently even in a difficult moment.

That costs more in the short term.

But it protects trust among those who stay. It protects reputation. It reduces resistance to transformation. It reduces the risk of internal decay.

That is the difference between cutting people and managing change as part of company architecture.

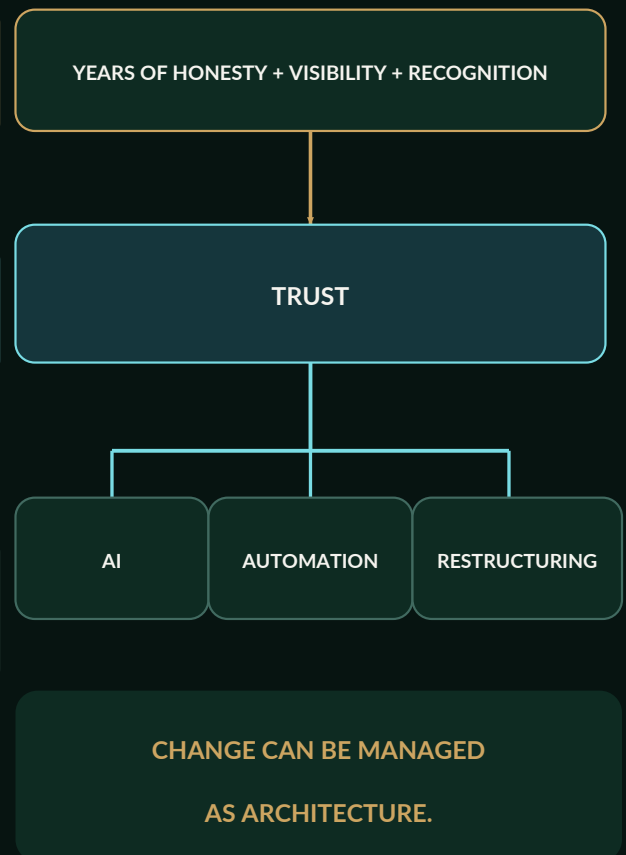
A well-designed human architecture has another effect.

The company stops being only an employer with job ads.

It becomes a place people want to enter.

Some companies do not need to post job offers because they already have a queue of candidates. People know from the market that it is worth working there. They know it not from employer branding campaigns, but from other workers.

Employer branding is what a company says about itself.



# OPERATIONAL REPUTATION

Operational reputation is what workers say about the company when HR is not listening.

That is a powerful strategic advantage.

A company that constantly has to buy people with ads, benefits and higher pay is often paying for its own weak human architecture. A company people wait to join has something stronger than a recruitment campaign.

It has a system people recognize as good.



**A SYSTEM PEOPLE RECOGNIZE AS GOOD.**

# THREE OPERATING MODES

Company architecture is not needed only when the company is already bleeding.

The best time to redesign a company is when it still has money, energy, people, reputation and time. Then architecture can extract hidden value, increase resilience, improve flows, remove friction, build better control and prepare the company for crises that have not arrived yet.

A company with visible problems needs stronger intervention. Loss sources must be identified, processes rebuilt, tools changed, roles redesigned, logistics improved, quality protected, communication corrected and reporting connected to reality.

A company on the edge of bankruptcy is a different mode.

In crisis mode, architecture is no longer advisory. It becomes command structure.

There is no room for politics, soft recommendations and “we will consider it.” Either the board gives real decision authority and executes the rescue architecture, or the company continues dying. Even then, sometimes it can be done with minimal resources: paper, free software, cheap tools and brutally simple processes.

But one thing must exist.

Execution discipline.

A company in crisis does not always need perfect tools.

It needs flawless operating logic on the resources it still has.

## HEALTHY

redesign while  
resources exist

## VISIBLE PROBLEMS

strong intervention

## EDGE OF BANKRUPTCY

command structure

**THE ARCHITECTURE CHANGES MODE.**

**EXECUTION DISCIPLINE DOES NOT.**

# THE GOAL IS REAL CONTROL

The goal of company architecture is not micromanagement.

The goal is real control.

Control does not mean the board manually steers every position, every machine and every person. Control means visibility, responsibility, measurability, clear flows, fast deviation detection and response capability.

A well-designed architecture should allow leadership to move from the level of the whole company down to a specific department, workstation, machine, process and value-loss point.

Vertical and horizontal communication should be designed so that reports at headquarters are not decoration, but a reflection of operational reality. With a properly designed system, reporting can show the performance of a specific workstation, a specific machine and a specific process with very high operational accuracy – even down to around 1% of operational throughput when measurement points and data validation are designed correctly.

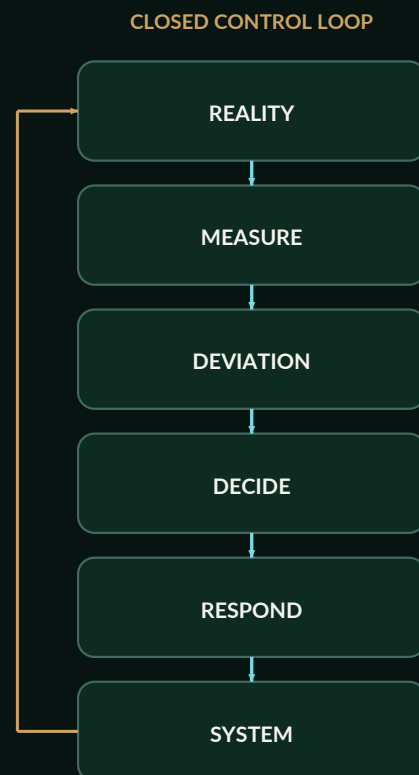
That is the difference between a dashboard and control.

A dashboard can show numbers.

Architecture makes sure the numbers are connected to reality.

The board does not need more dashboards.

It needs a company designed so that dashboards show reality.



**A DASHBOARD SHOWS NUMBERS.**

**ARCHITECTURE CONNECTS THEM TO REALITY.**

# CONSULTING VS ARCHITECTURE

This is the difference between consulting and architecture.

Many consulting interventions optimize the reporting layer.

A systems architect redesigns the reality layer.

Consulting can describe the problem. Architecture designs the system that stops producing it.

Consulting can improve a KPI. Architecture checks whether the company is becoming weaker under a better KPI.

Consulting can create a process. Architecture creates the operating structure in which procedures, people, tools, software, logistics, quality, data, decisions and reporting work as one system.

Consulting can say the company has a communication problem. Architecture designs the information flow so communication is embedded into execution.

Consulting can create a presentation about culture. Architecture designs the conditions in which a human being feels seen by the company.



# THE CORE OF MY WORK

That is the core of my work.

I see the layers of the system and I design coherent flows inside each layer and between layers, so the company does not operate on momentum, but as a consciously designed organism.

Procedures must be clear.

Processes must reduce friction.

Tools must support work.

Software must execute the architecture.

People must feel they are part of the company.

The board must have real control over the system.

Not over the presentation.

Over the company.

Consulting can make the company look better in a report.

Architecture makes the company work better in reality.

PROCEDURES

CLEAR

PROCESSES

REDUCE FRICTION

TOOLS

SUPPORT WORK

SOFTWARE

EXECUTE ARCHITECTURE

PEOPLE

PART OF COMPANY

BOARD CONTROL

OVER THE SYSTEM

**NOT OVER THE PRESENTATION.**  
**OVER THE COMPANY.**

---

# ARCHITECTURE MAKES THE COMPANY WORK BETTER IN REALITY.

The report is a representation. The company is the operating system underneath it.



MATEUSZ CHRZANOWSKI

SYSTEM OF SYSTEMS ARCHITECT

COMPANY ARCHITECTURE PAPERS

*Chrzanowski*