Selmo

Intelligent machine behavior is a decision

Selmo Method™

Formally define machine behavior, make it clearly decidable and traceable.
From BlackBox thinking to WhiteBox responsibility.

WhiteBox Engineering: Behavior is defined. Responsibility is clear.

BlackBox systems conceal behavior in code – risks only become visible during operation. WhiteBox makes machine behavior explicit, verifiable, and decidable before programming. The Selmo Method shifts the focus from hope to evidence – and makes machine behavior controllable.

Roadmap to WhiteBox

01 – Decide machine behavior

Define and decide machine behavior together before technology and code exist.

02 – WhiteBox emerges

Behavior is formally, unambiguously, and completely modeled and specified.

03 – Safety before implementation

Simulation verifies behavior before commissioning – risks become visible before they become costly.

04 – Technology follows the decision

Code is derived from the model – deterministic instead of interpreted.

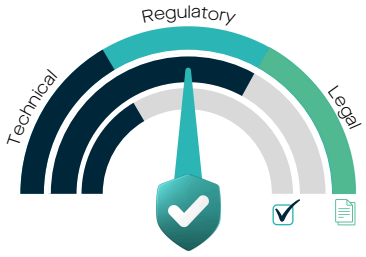
05 – WhiteBox as an asset

Behavior remains auditable and stable – even during lifecycle changes.

Machine behavior under control. Your machine is now a premium asset.

Risk becomes controllable

Governance before commissioning



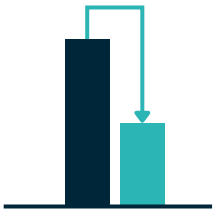
Technical, regulatory, and legal risks are managed in the model – long before physical operation.

BlackBox vs. WhiteBox Costs

WhiteBox significantly reduces downtime, commissioning time, and maintenance. BlackBox shifts costs from the project phase into operation – and increases the risk of later escalations.

120 % ROI

Structural cost reduction across the lifecycle.



WhiteBox Engineering is governance

Selmo decides behavior before code is written – and turns machines into transparent, auditable assets.

WhiteBox is not an option.
It is the prerequisite for controllable operation.

Undefined machine behavior is not an engineering problem. It is a decision-making problem.

For decades, industrial machines have been built to function – not to make their behavior understandable.

Behavior emerges from code, decisions made during commissioning, suppliers' subjective interpretations, and subsequent corrections. That works for a while – until problems suddenly occur.

What is missing is formal decision-making about behavior.

This is what happens when machine behavior is not explicitly defined:

- States exist that no one can fully describe
- Transitions occur that are never formally verified
- Changes accumulate without validation by stakeholders

The result is not just inefficiency. The result is structural uncertainty.
Decisions regarding safety, performance, compliance, and future scalability are made without clarity on how the machine is actually intended to behave.

If behavior cannot be formally decided, the machine cannot be operated responsibly.

If no one can explain why the machine behaves the way it does, no one truly has it under control.

Many organizations assume they control machine behavior. In reality, control is often distributed.

Behavior is shaped by teams, suppliers, and continuous changes. Responsibility exists, but without a clear owner.

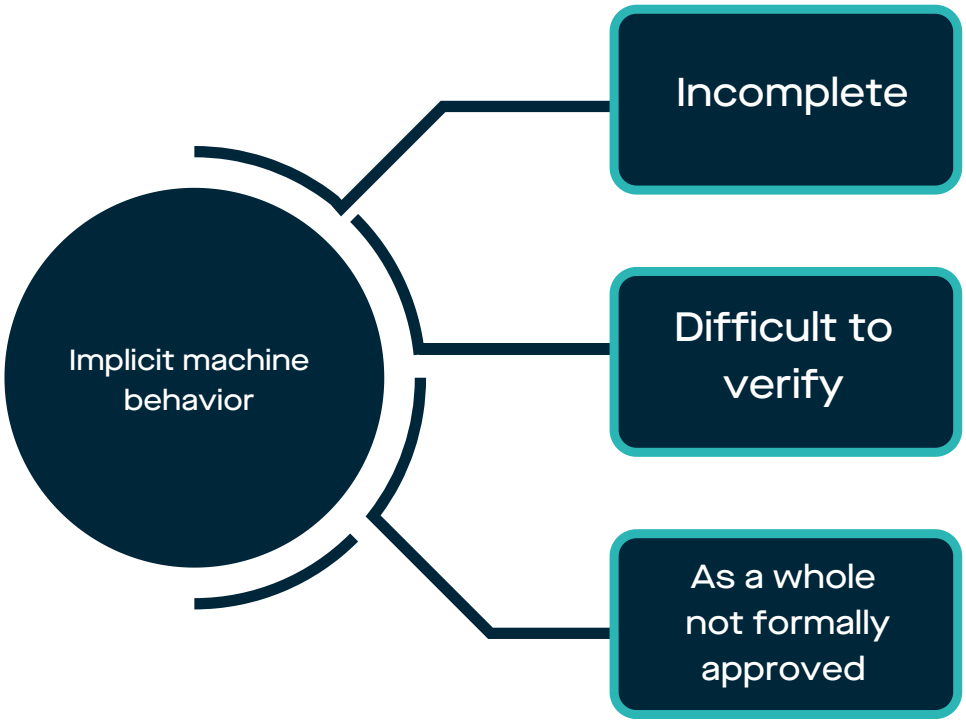
Machine behavior must be decided before implementation

In classical engineering, machine behavior is defined too late.

How a machine behaves only emerges during programming, commissioning, and iterative troubleshooting – after mechanical, electrical, and contractual decisions have long been made.

At that point, behavior can no longer be decided; it can only be adjusted.

Even the most experienced engineers, the best tools, and the strictest reviews cannot change this structural fact: code describes how a system is executed, not why it behaves the way it does. As long as behavior is implicitly defined through implementation, it remains:



If machine behavior is only decided in code, it can no longer be defined in advance – it is only discovered during operation.

The Selmo Method™ defines exactly this decision layer.

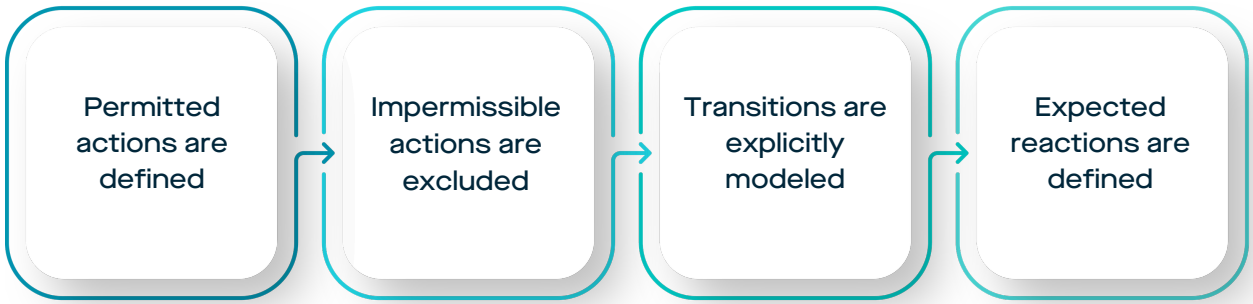
It establishes a formal, deterministic model in which machine behavior is fully described, validated, and approved before execution logic is generated.

Selmo does not improve programming. It decouples behavior from code.

Formal behavior means that nothing remains undefined.

Formally defined machine behavior means that the machine can only assume explicitly defined states.

For each state:



There are no hidden paths. No implicit assumptions.
No behavior that “usually” works.

Every transition is intentional.
Every state is verifiable.
Every behavior is explainable.

This is what determinism means – not predictability through experience, but predictability through definition.

If a state cannot be formally described,
it must not exist.

Decided behavior must be continuously verified.

A formally defined behavior model is only meaningful if it remains valid during operation.
Machines change:

Without regular verification, even clearly defined behavior gradually diverges from what was originally intended.

For machines to behave truly deterministically, it must be continuously checked whether the current state still corresponds to what was previously defined.

Deviations become visible immediately – not weeks later during troubleshooting.

This makes operation transparent:
Behavior is not assumed or inferred, but continuously verified and confirmed.

Behavior is only deterministic
if deviations are detected immediately.

Most operational problems are symptoms, not causes.

Across industrial organizations, the same frustrations keep recurring:

> "We don't really know what the machine is doing."

> "The behavior changes depending on who modified the code."

> "Small changes break something elsewhere."

> "Diagnostics are unclear and inconsistent."

These are not isolated problems. They are recurring symptoms. What they share is not insufficient execution, but the absence of a defined reference for how the machine is intended to behave.

Without a formally decided model:

- Deviations remain undetected
- Behavior is interpreted, not verified
- Troubleshooting replaces understanding

The system does not fail randomly. It behaves this way because its behavior was never explicitly decided.

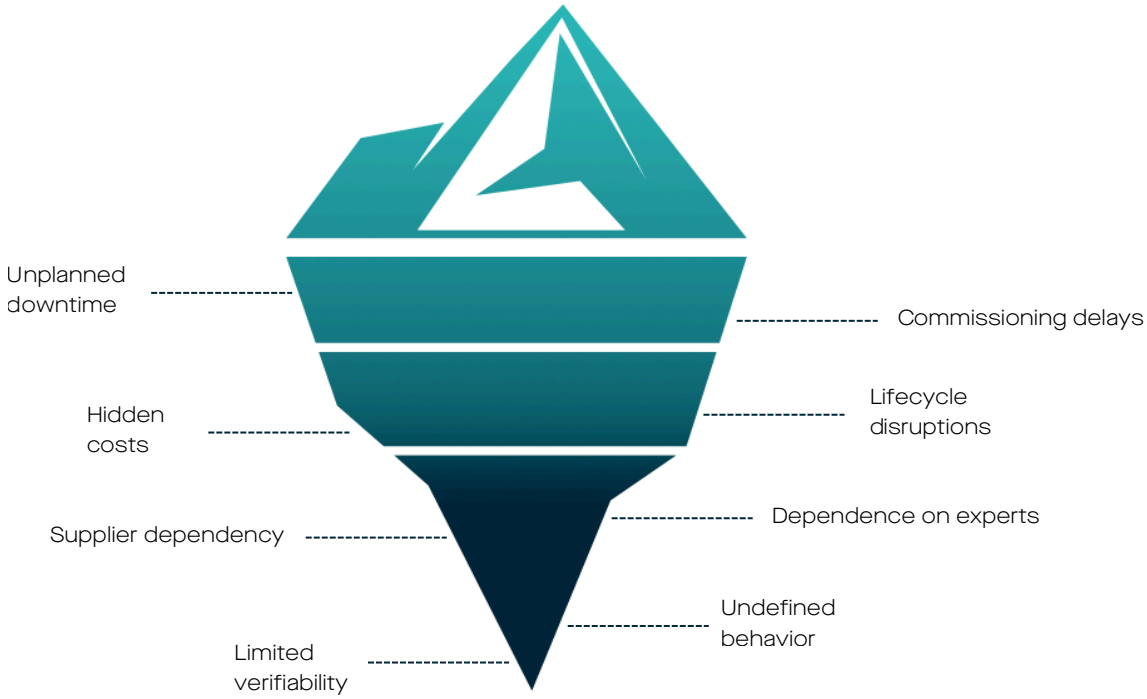
If deviations are not structurally visible,
problems will always appear as surprises.

Undetected deviations create hidden risks

Across industrial organizations, the same frustrations keep recurring. When deviations from intended behavior go unnoticed, their impact accumulates.

In operation, this appears as:

- Unplanned downtime
- Fragile processes
- Inconsistent performance
- Growing dependence on individual experts



Structurally, the consequences go deeper.

Decisions on safety, compliance, throughput, and future changes are made without a reliable foundation for machine behavior.

Risks are not assessed; they are absorbed. Over time, organizations adapt to uncertainty instead of eliminating it.
What feels like stability is often just the absence of an acute failure.

Risk does not arise from deviation itself,
but from deviation that remains undetected.

Premium performance requires clearly defined behavior.

Industrial organizations invest heavily in performance, quality, and availability.

They specify tolerances, cycle times, safety levels, and contractual commitments – often down to the smallest detail.

Yet the most fundamental aspect often remains implicit: how the machine is allowed to behave.

Without a formal decision about behavior:

Performance depends on interpretation

Stability depends on individuals

Quality declines over time

Premium execution cannot be built on undefined behavior. It requires behavior that:

is explicitly defined and decided

is structurally enforced

is continuously verified

Only machines with decided and defined behavior can be delivered as premium assets.

A complete system for decidable machine behavior

The Selmo Method™ integrates eight structural dimensions into a single deterministic framework:

Deterministic state architecture

All states and transitions are conflict-free and explicitly permitted.

Upstream behavior verification

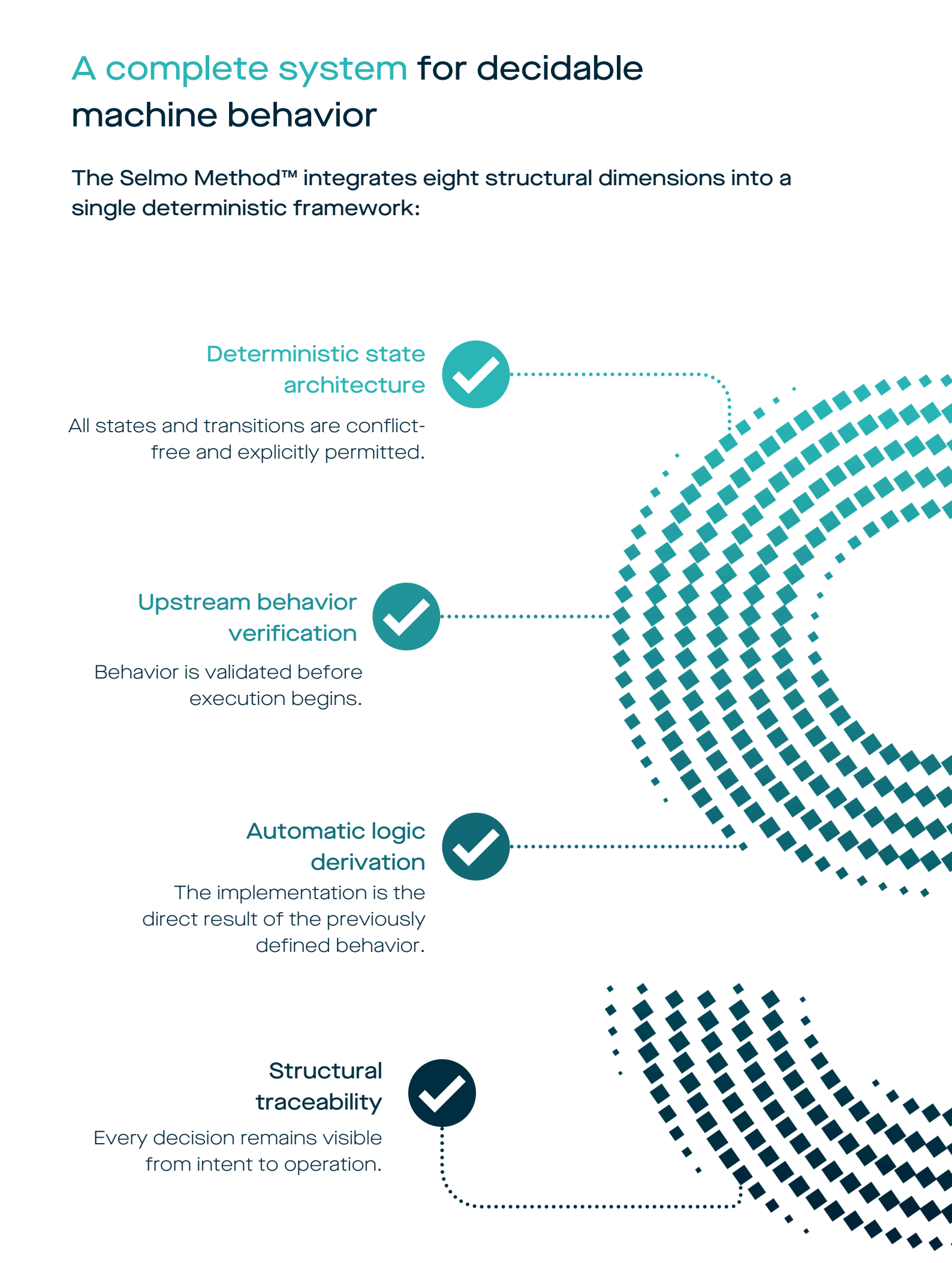
Behavior is validated before execution begins.

Automatic logic derivation

The implementation is the direct result of the previously defined behavior.

Structural traceability

Every decision remains visible from intent to operation.



Decidable machine behavior cannot be achieved through isolated measures. It requires a closed, coherent system that defines behavior, validates correctness, and preserves decisions across the entire lifecycle. This is exactly what the Selmo Method™ is built for.

Formal behavior modeling

Machine behavior is mathematically defined, not informally described.

Lifecycle stability

Behavior remains correct despite changes, updates, and extensions.

Deterministic operational data

Every signal has context, meaning, and reference.

Organizational standardization

A shared language for machine behavior across teams and suppliers.

Only a consistent system ensures that machines behave as defined and decided – across projects, suppliers, and years.

Selmo Method™

A formal decision standard for machine behavior

The Selmo Method™ establishes a formal decision framework in which machine behavior is fully defined, reviewed, and validated before implementation begins.

It separates behavioral decisions from technical execution.

This turns machine behavior into a structural, verifiable, and auditable asset.

Description of our system

The Selmo Method™ is based on a deterministic system.
It is characterized by:

Formal behavior modeling
Every state, every action, and every transition is explicitly defined.

Deterministic state architecture
The model is fully consistent, without hidden paths or undefined states.

Behavior verification before execution
The complete behavior is validated as correct before it is implemented in control logic.

Structural traceability
Every decision is traceable from intent through execution to operation.

Standardized behavior across organizations
The same behavior language applies across teams, sites, and suppliers.

Selmo does not optimize behavior.
It decides it.

Make the difference between conventional machines and Selmo-based machines visible and verifiable

P

Process

Describe what the system is intended to do as a sequence of states, actions, and expected reactions.

T

Technology

Define how each expected reaction can be physically detected using sensors and actuators.

F

Function

Link the defined process logic (P) with the measurable technology (T) to ensure feasibility.

From BlackBox thinking to WhiteBox responsibility.

Transform a risky machine operating as a “BlackBox” into a transparent “WhiteBox” machine, where all behavior is explicit and traceable.

Automated

Code generation

The model is automatically translated into PLC code (e.g., ladder logic), eliminating manual programming.

Commissioning

& Validation

Since commissioning is not for problem-solving, it is a formal alignment of the model with the real hardware.

Formal

Modeling

The PTF results are transferred into a formal process model that becomes the single authoritative reference.

Live

“WhiteBox” Operation

The system continuously operates the model-defined state using real system data and detects any deviation immediately.

One structure. Two fundamentally different realities.

Machines may appear identical externally. Their structural logic is not.

	Conventional machines	Selmo machines
Behavior	Behavior emerges from implementation	Behavior is decided before implementation
States	States are implicit or incomplete	All states and transitions are explicit
Deviations	Deviations are detected late or indirectly	Deviations are detected immediately
Knowledge	Knowledge resides with individuals	Knowledge is structurally embedded
Changes	Changes lead to uncertainty	Changes are verified against the model
Quality	Behavior quality declines over time	Behavior quality is maintained long term

The difference is not in components or code, but in how behavior is defined, verified, and preserved.

Selmo does not improve execution.
It changes the underlying structure.

Clarity before implementation.
Structure before execution.

Decidable machine behavior does not emerge by chance. It is the result of a clearly defined decision process that leads from intent to execution while preserving structure and responsibility.

The Selmo Method™ establishes this process.

The Selmo decision process

PTF — Process · Technology · Function

The intended machine behavior is defined before a solution is implemented. Processes are clarified, technologies evaluated, and functions explicitly decided. This creates a shared, verifiable understanding: “Yes, this is the machine we want to build.”

Formal behavior modeling

The decided behavior is transferred into a formal, state-based model. Every state, action, and transition becomes visible and verifiable – like an electrical schematic for behavior. There are no hidden assumptions. No undefined states.

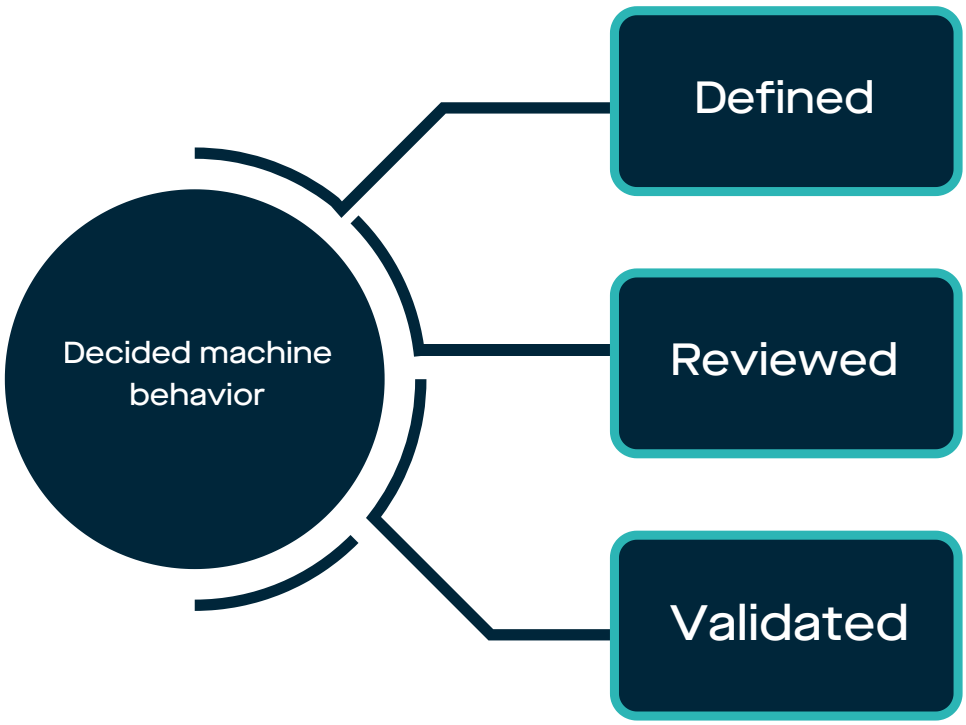
Deterministic implementation mapping

From the validated model, execution logic is structurally derived. Implementation follows the decision, not interpretation. The resulting control logic is consistent, standardized, and ready long before commissioning.

Implementation becomes the execution of a decision, not the place where the decision is made for the first time.

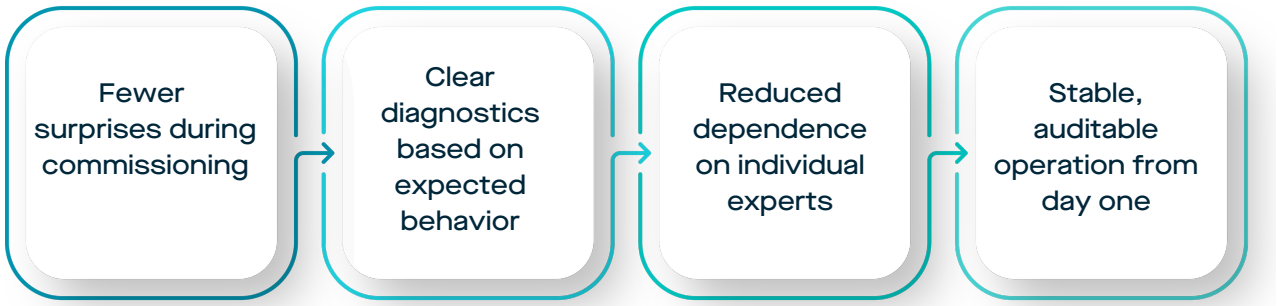
When behavior is decided, integration becomes predictable.

Integration and commissioning are not challenging because machines are complex, but because machine behavior only becomes visible at that stage. With Selmo, this changes fundamentally.



Integration becomes a continuous alignment between reality and a known structure. Virtual validation confirms correctness at an early stage. On-site commissioning focuses on alignment, not interpretation. The result is not only speed, but clarity. Engineers work with a transparent model. Operators gain continuous insight into machine state and behavior.

Operational impact



Commissioning is not a search for behavior, but its confirmation.

Decide with clarity, before you commit.

Working with Selmo is intentionally structured to remove uncertainty already at the decision stage.

Before implementation begins, we establish a shared understanding of how the machine is intended to behave – and what it must not do.

This ensures that decisions are made consciously, transparently, and without pressure.

Clear steps toward premium, safe, and intelligent machines

01

Strategic clarification:

The first step is a focused strategy session. It is not about sales, but about clarity:

- Where machine behavior is currently implicit
- Where decisions are made without transparency
- And where structural risk exists

On this basis, we jointly assess whether Selmo fits your context.

02

Potential analysis & PTF:

In the next step, the intended machine behavior is formally analyzed using the PTF framework.

Process, technology, and function are aligned to a shared, clear reference.

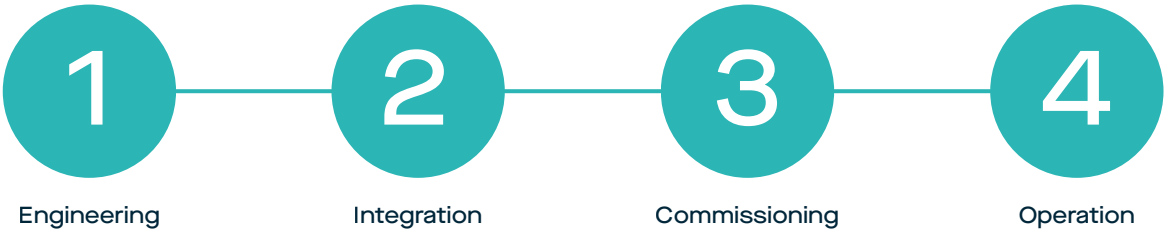
The result is a defined decision point: machine behavior is consciously and explicitly determined.

“Yes — this is the behavior we want to stand for.”

No behavior is implemented before it has been consciously decided.

From defined and decided behavior to controlled execution.

Once machine behavior has been formally decided, implementation follows a clear and controlled path. The behavior model becomes the single authoritative reference for all subsequent steps:

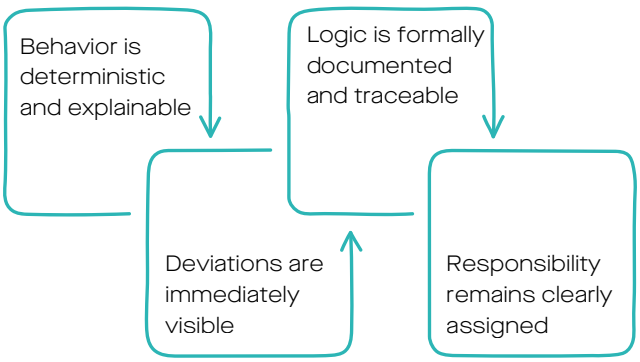


Execution logic is derived from the model, not interpreted from requirements. The result:

- Integration aligns with known behavior
- Commissioning confirms correctness
- Changes remain structurally validated

Not just a running machine, but a controlled system.

Operational outcome



The machine goes into operation as a premium asset. This ensures trust – not only at initial operation, but over the entire lifecycle.

A Selmo-based machine does not rely on hope.
It relies on decided structure.

Why AI requires structured decision foundations

Artificial intelligence does not fail due to missing algorithms. It fails due to missing structure. AI systems require:

- Clearly defined states
- Explicitly defined transitions
- Reliable, contextualized data

Without formally decided machine behavior, data loses meaning, events become ambiguous, and outcomes are no longer explainable. In such environments, AI decisions cannot be validated and therefore cannot be deployed responsibly.

The Selmo foundation

By transforming machine behavior into a formal, deterministic structure, Selmo provides what AI fundamentally requires:

- Clear operational context
- Explainable system states
- Traceable cause-and-effect relationships

AI optimizes based on clear specifications; it does not decide blindly.

AI is only as trustworthy as the structure on which it operates.

Deciding on machine behavior is no longer optional.

Industrial organizations reach a point where implicit machine behavior is no longer sufficient.

Complexity increases. Responsibility becomes clearer. Expectations for transparency, safety, and explainability continue to rise.

In this environment, it is no longer enough to rely on assumptions about machine behavior. It requires:

Explicit decision

Formal definition

Continuous verification

Selmo is for organizations that choose clarity over assumptions and structure over interpretation.

Positioning

Selmo is not for everyone. It is for organizations that take it seriously.

It is for organizations that:

- Want to understand what their machines actually do
- Take responsibility for that behavior
- And preserve this clarity across teams, suppliers, and years

Clarity is not a feature.
It is a decision.

Selmo

Clarity is not optional.

Are you ready to structurally anchor performance improvement?

6 clearly measurable effects resulting from decided machine behavior

Up to 40% shorter project timelines.
When machine behavior is decided before programming begins, late iterations and interpretation loops are eliminated. Projects become predictable. Changes remain controllable. Time-to-market is measurably reduced.

OEE of 85–95% sustainably achievable.
Higher availability, reproducible performance, and fewer disruptions make OEE controllable. Not estimated. Not explained. Structurally secured.

Up to 70% fewer unplanned downtimes.
Deterministically defined machine behavior prevents structural deviations. Errors become immediately visible. Diagnostics are based on reference — not interpretation.

Up to 50% faster commissioning.
Pre-validated, deterministic behavior models structurally reduce uncertainty during commissioning. Acceptance becomes clearer, faster, and reproducible — not dependent on experience.

Up to 20% higher output.
Stable, deterministic processes eliminate recurring losses and bottlenecks. Performance gains result not from pressure — but from structure.

Up to 50% lower maintenance effort.
Transparent states and unambiguous logic significantly reduce troubleshooting time. Deviations are instantly visible. Maintenance becomes structured, targeted, and data-driven — instead of reactive.

If you pay for premium performance, shouldn't your machines deliver premium behavior?

Make a structural decision.

Schedule a strategy session now.

Selmo Technology GmbH
einfach@selmo.at
+43 3136 20755

selmotech.com