Data Science Process:

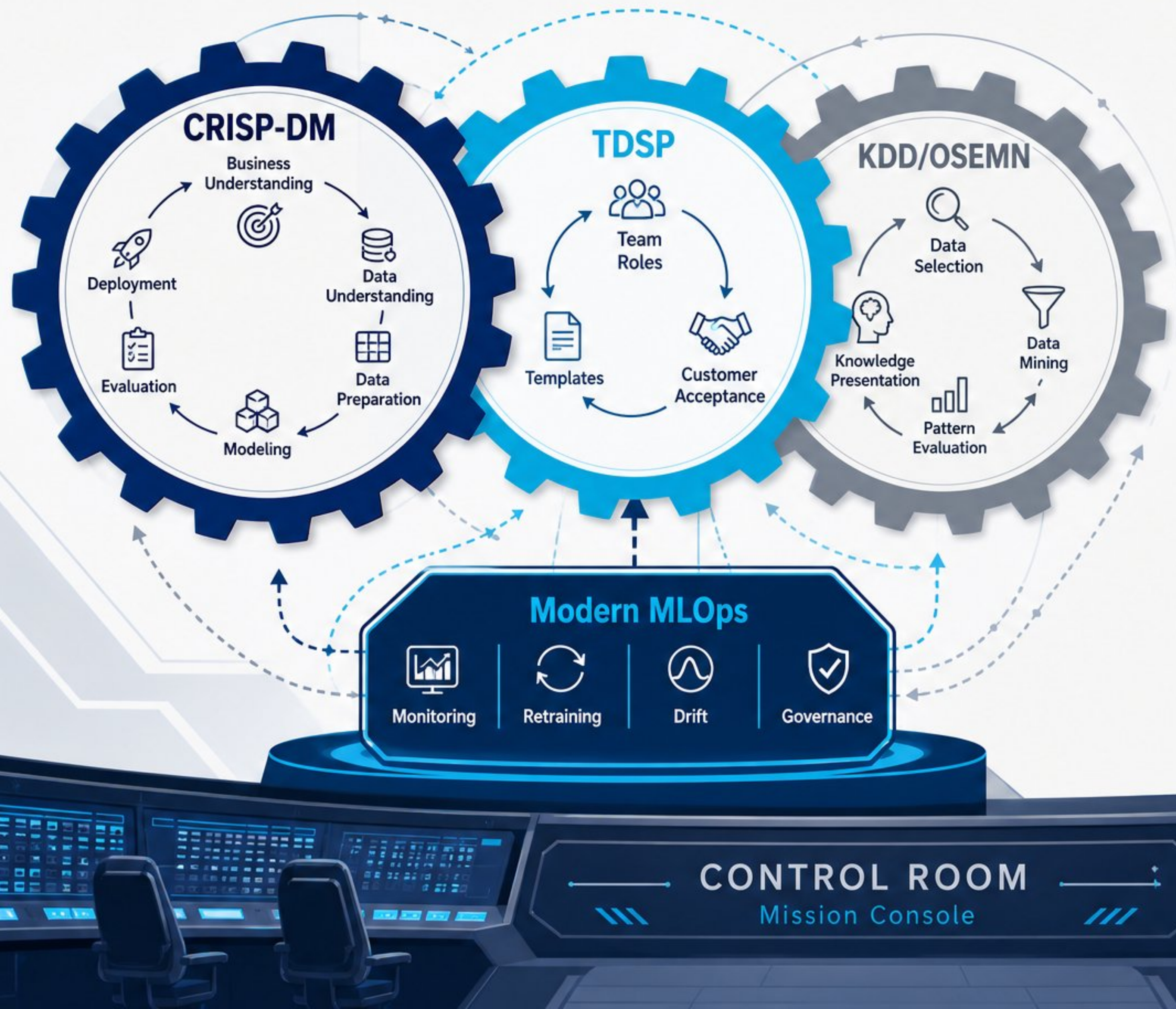
Stages, Roles, and Deliverables

- - Structured process separates shipping models from stagnant notebooks
- - Map the full lifecycle: problem framing to monitoring and retraining
- - Designed for learners, data teams, and technical managers
- - Master stage mapping, role assignment, and stage-specific deliverables

Why Process Discipline Beats Ad-Hoc Analysis

- 70–87% of data science and AI initiatives fail to deliver value
- Root cause: broken workflow, unclear ownership, poor data foundations
- Structured process improves reproducibility, risk control, and trust
- Process is a shared language across scientists, engineers, and leaders

Lifecycle Frameworks: CRISP-DM, TDSP, KDD, and Beyond

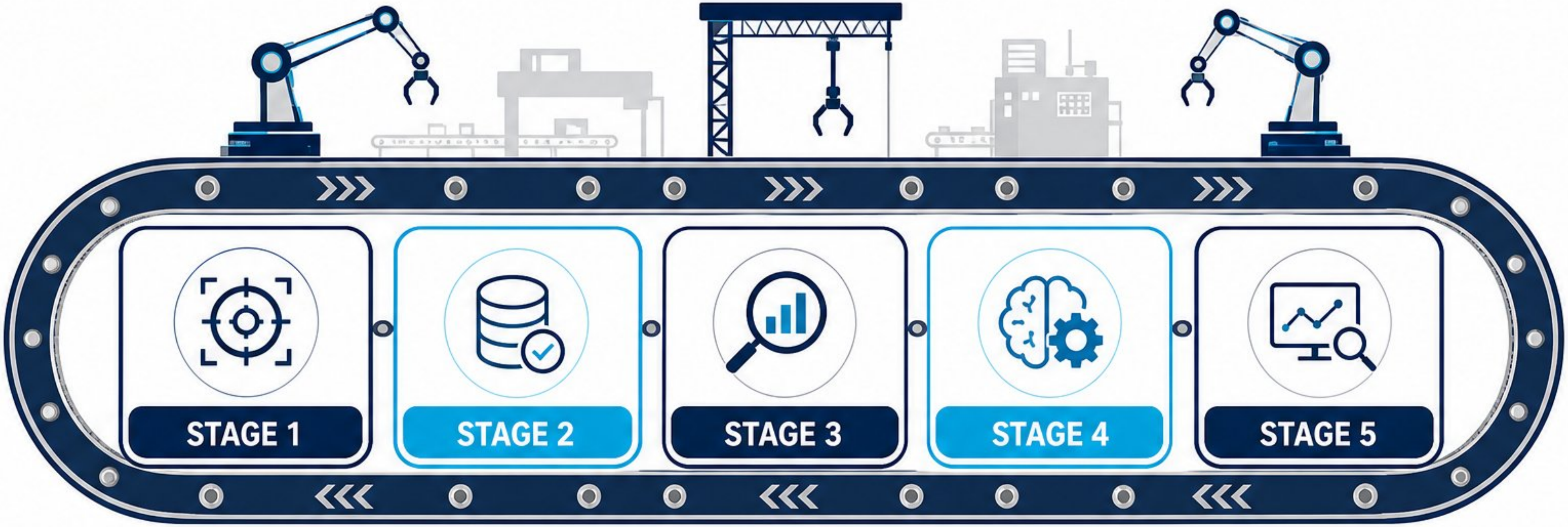


- **CRISP-DM:** six iterative phases, de facto industry reference
- **TDSP:** adds team roles, templates, customer acceptance
- **KDD/OSEMN:** earlier models for knowledge discovery
- **Modern MLOps:** monitoring, retraining, drift, governance

Choosing and Adapting a Framework for Your Context

- **CRISP-DM:** flexible, tool-agnostic mental model for experienced teams
- **TDSP:** added structure, roles, and templates for newer teams
- Neither covers modern MLOps; pair with monitoring and maintenance
- Avoid waterfall rigidity; plan for iteration from day one

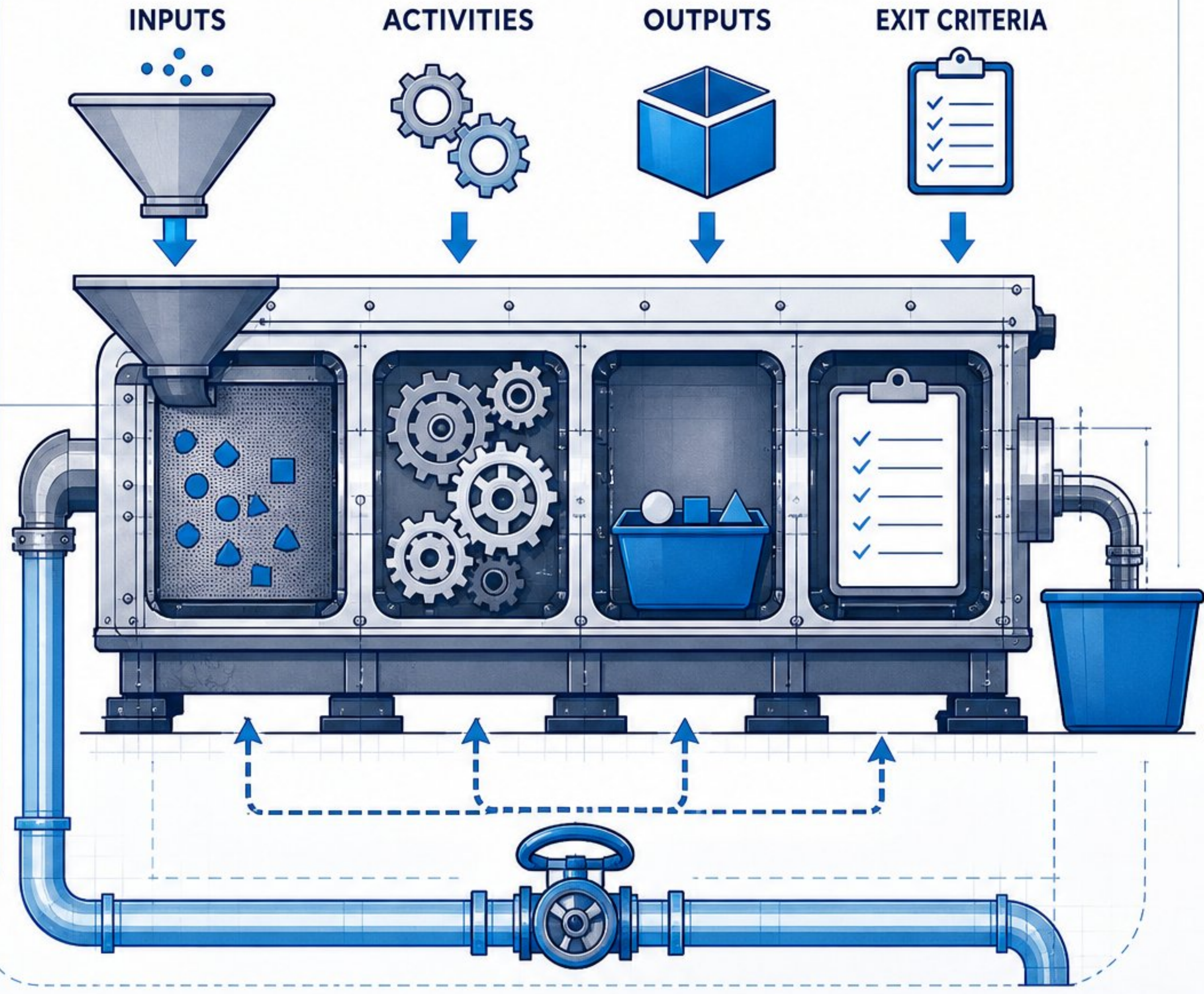
Core Stages: From Problem Framing to Monitoring



STAGE 1	Stage 1: Problem framing, success metrics, and project charter
STAGE 2	Stage 2: Data acquisition, profiling, validation, and data contracts
STAGE 3	Stage 3: EDA, feature engineering, and preparation for modeling
STAGE 4	Stage 4: Modeling, evaluation, interpretation, and decision to deploy
STAGE 5	Stage 5: Deployment, monitoring, governance, and continuous improvement

Stage Inputs, Activities, Outputs, and Exit Criteria

- **Each stage:** clear inputs, activities, outputs, exit criteria
- **Example:** success metrics before data work; data quality report before modeling
- **Example:** model card before deployment
- **Iteration is normal:** later stages often trigger reframing or new data
- **Stall points:** endless EDA, premature modeling, unclear deployment ownership



Roles in a Data Science Team

- **Data Scientist:** problem framing, modeling, validation
- **Data Engineer:** pipelines, data quality, and availability
- **ML Engineer/MLOps:** deployment, monitoring, and retraining
- **Data Analyst:** dashboards and business insight reporting
- **Product Owner:** prioritization and alignment with business goals
- **Analytics Translator:** connecting technical results to business value

RACI Mapping Across Lifecycle Stages

- RACI: Responsible (does work), Accountable (owns outcome), Consulted (two-way input), Informed (one-way updates)
- Every activity needs exactly one Accountable; multiple A's cause decision gridlock
- Typical handoff: Data Science → ML Engineering for deployment; Platform Owner accountable for incidents
- Keep Consulted roles lean (2-3) to maintain speed and clarity
- Treat RACI as a living document, reviewed at project milestones and after incidents

LIFECYCLE STAGES

	Problem Definition	Data Preparation	Model Development	Model Validation	Deployment	Monitoring & Operations
Data Scientist	R	R	R	C		
ML Engineer					A	R
Platform Owner						A
Data Steward	C	R	C	C	C	C
Compliance Analyst	I	I	I	I	I	I

R Responsible (does work) · **A** Accountable (owns outcome) · **C** Consulted (two-way input) · **I** Informed (one-way updates)

Deliverables by Lifecycle Stage

Framing:

- Framing: problem statement, success metrics, project charter, data inventory

Data:

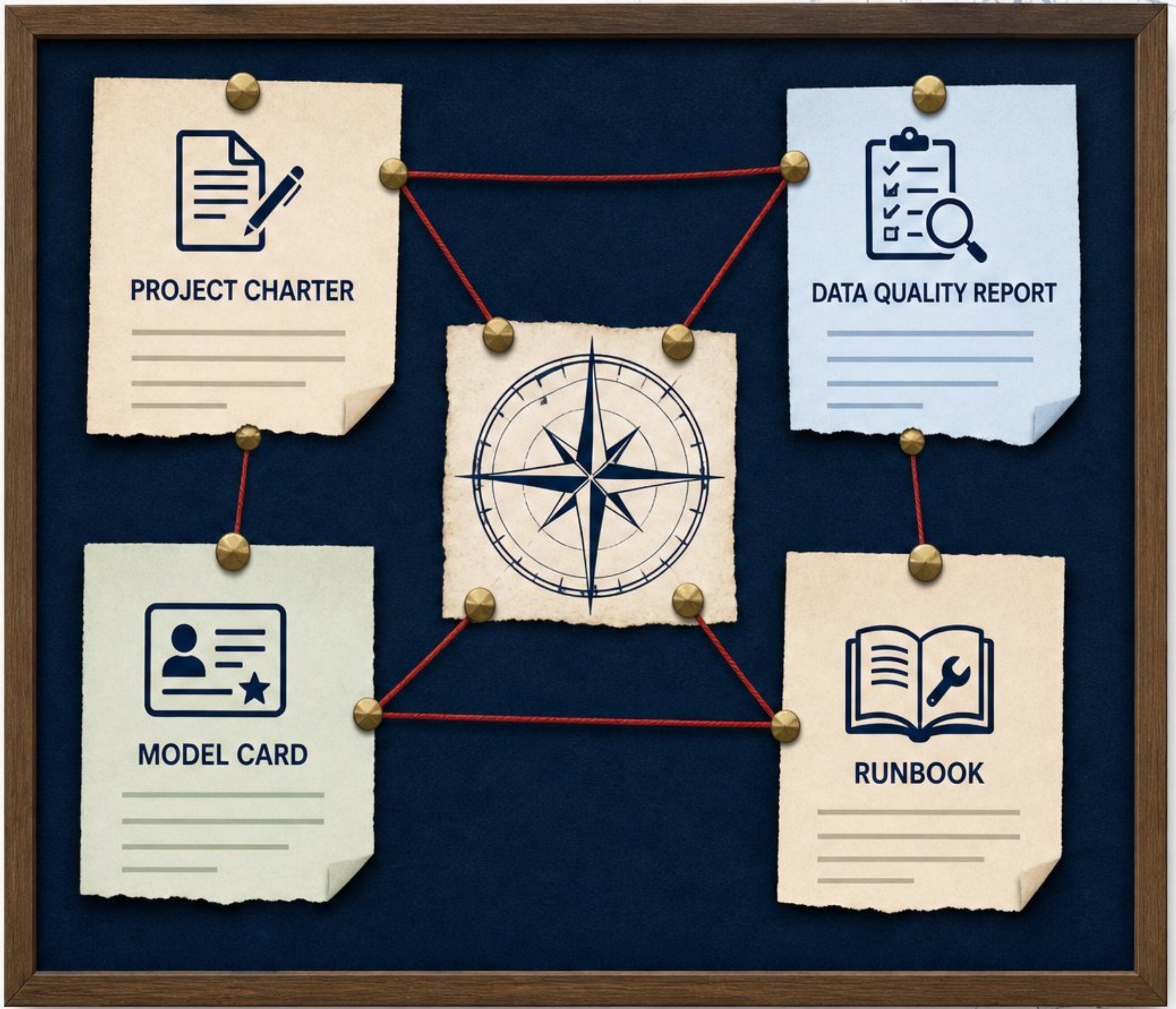
- Data: dictionaries, quality reports, lineage documentation, pipeline artifacts

Modeling:

- Modeling: EDA reports, feature specs, experiment logs, model cards, evaluation reports

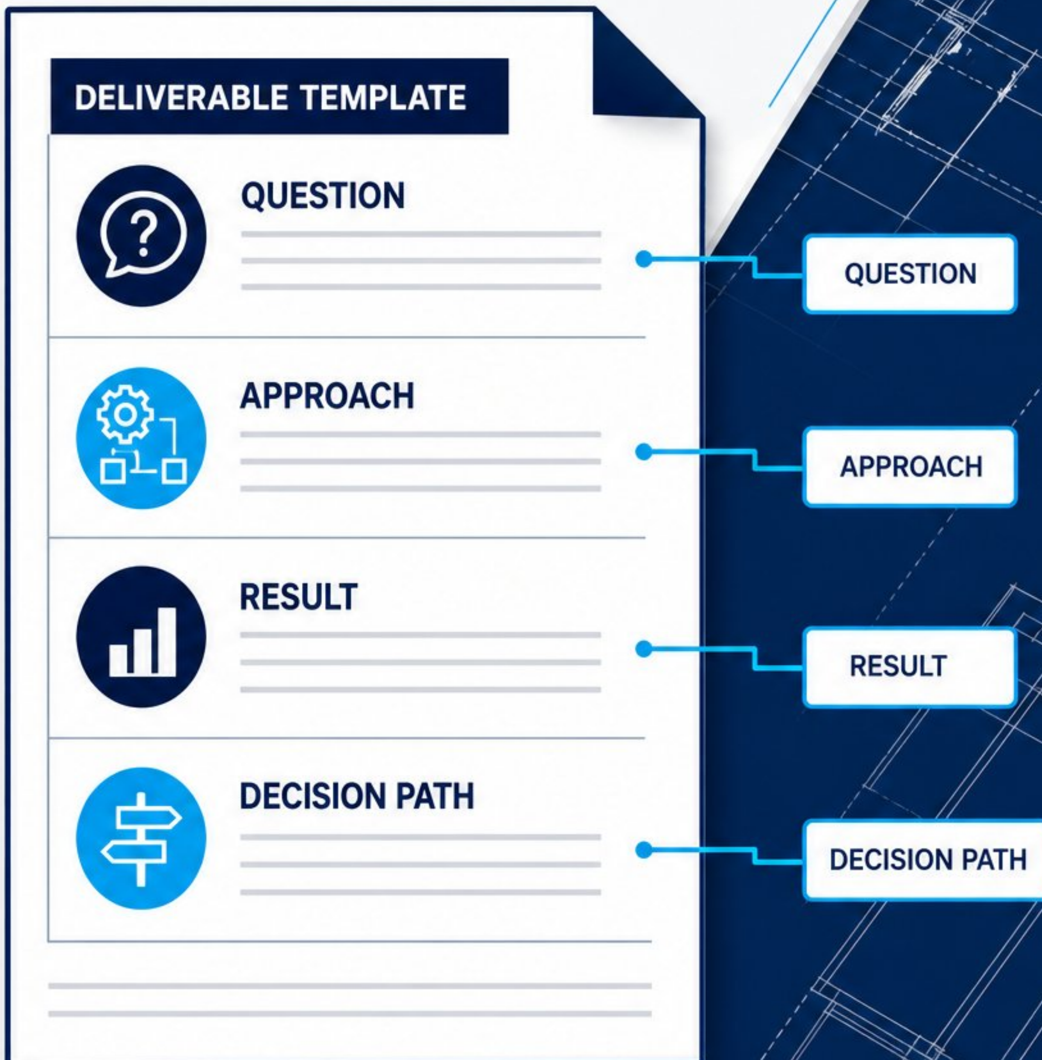
Post-deployment:

- Post-deployment: runbooks, monitoring dashboards, drift alerts, maintenance docs



What Makes a Deliverable Usable

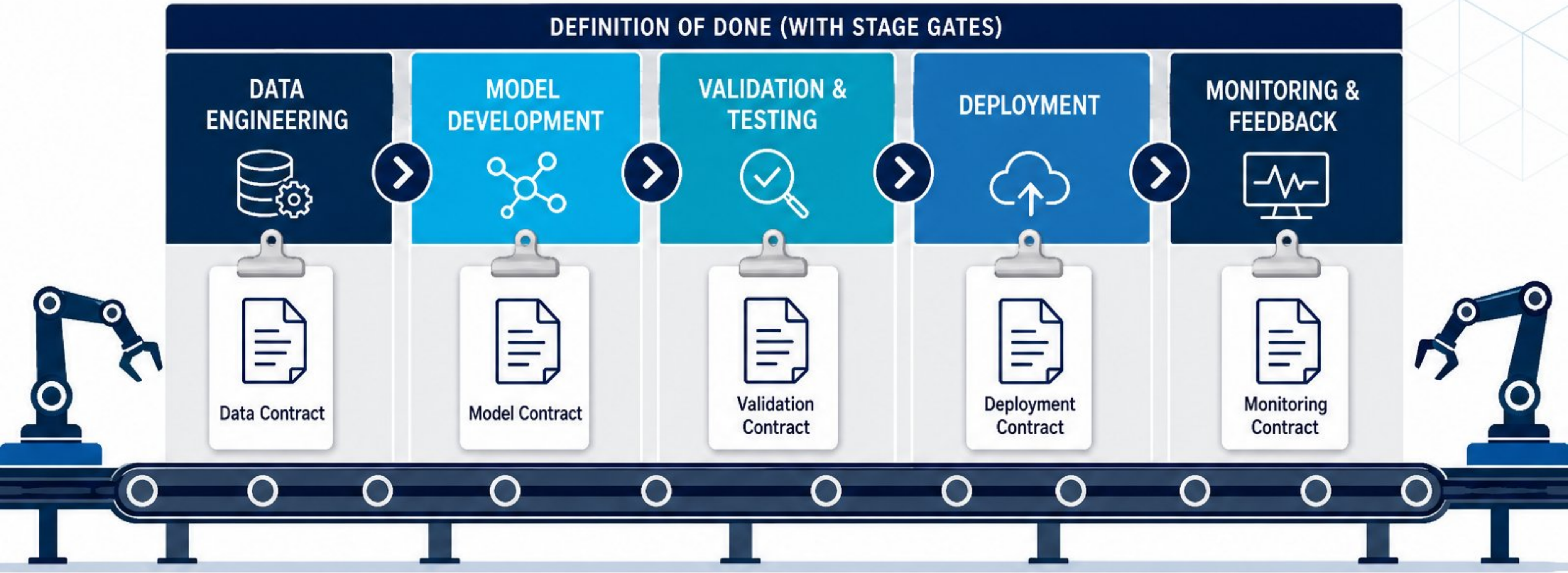
- Deliverables must serve both technical and business stakeholders
- State the question, approach, result, and decision path clearly
- Include what didn't work—honest limitations build trust
- Use templates for reproducibility and auditability
- Typical gap: code without usable documentation blocks handoff



Process in Practice for Learners

- Build full lifecycle projects: problem statement to deployment
- Minimum deliverables: clear README, honest evaluation, demo
- Explain interview projects: problem, approach, tradeoffs, limitations
- Avoid skipping problem framing and deployment as an afterthought

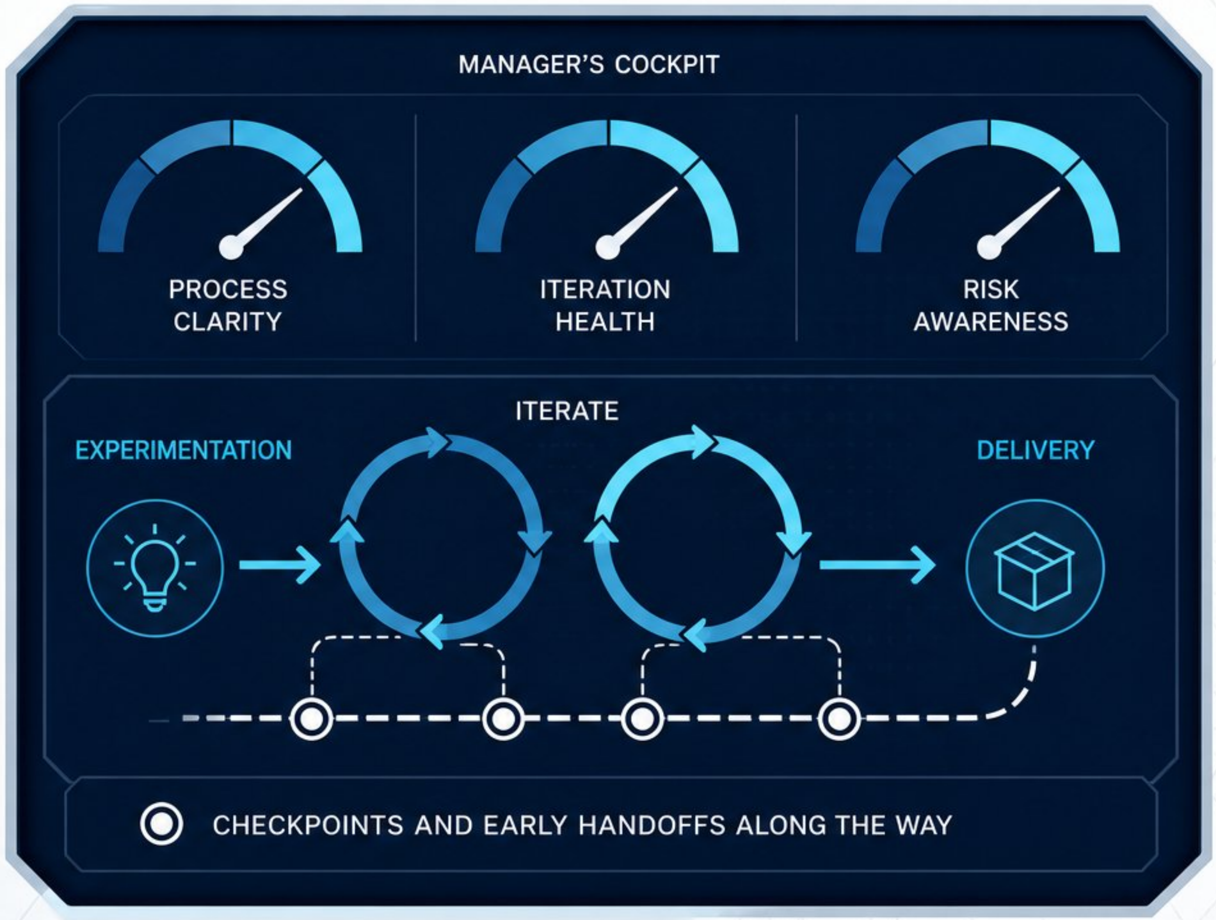
Process in Practice for Working Data Teams



- Embed process into sprint cadence; stage gates in definition of done
- Explicit handoffs between data engineering, modeling, and deployment
- Track progress with stage-specific metrics, not just model accuracy
- Use retrospectives to interrogate specific failure patterns
- Enforce gated progression reviewed jointly by technical and business owners

Process in Practice for Technical Managers

- Anchor planning to business objectives and measurable success metrics
- Set realistic timelines without waterfall rigidity; embrace iteration
- Balance experimentation with delivery via checkpoints and early handoffs
- Track progress and risk with stage metrics, not just model accuracy
- Coach teams with process as structure, not constraint



Common Pitfalls and Process Maturity

HAZARDS

- Frequent pitfalls: modeling before problem definition, ignoring data quality, and building deliverables nobody uses
- Other pitfalls: unclear ownership, no monitoring, and treating deployment as an afterthought

PROCESS MATURITY: HEAT ZONES

AD HOC → REPEATABLE → DEFINED → MANAGED → OPTIMIZED

- Process maturity levels: ad hoc, repeatable, defined, managed, and optimized
- Assess maturity across dimensions including reproducibility, evaluation, deployment, observability, and governance

Actionable Next Steps and Maturity Roadmap

Learners

- Learners: one end-to-end project with documented reasoning

Teams

- Teams: refine RACI, define deliverables, make handoffs explicit

Managers

- Managers: assess maturity, prioritize low-cost high-impact fixes

- First steps: standardize problem framing and add data quality checks

PersonWise.

PersonWise • AI digital-human course platform

Scan the code or click anywhere to watch this course live, with voice Q&A

personwise.ai/t/811d8468/public