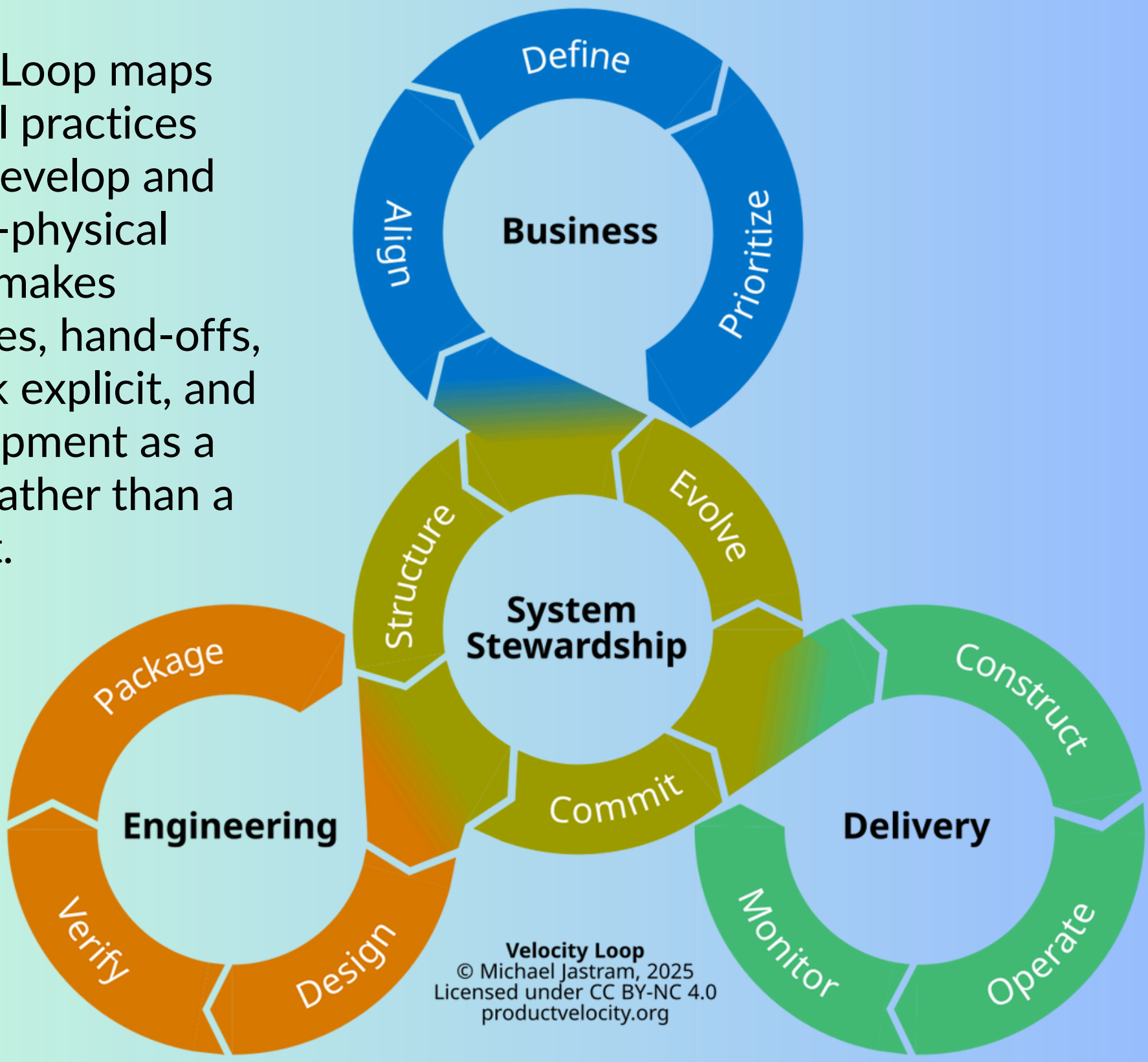
The Velocity Loop

A system-level model for continuous development of complex products

The Velocity Loop maps four essential practices required to develop and evolve cyber-physical products. It makes responsibilities, hand-offs, and feedback explicit, and treats development as a closed loop rather than a linear project.



Velocity Loop

productvelocity.org

www.productvelocity.org

Book by Michael Jastram

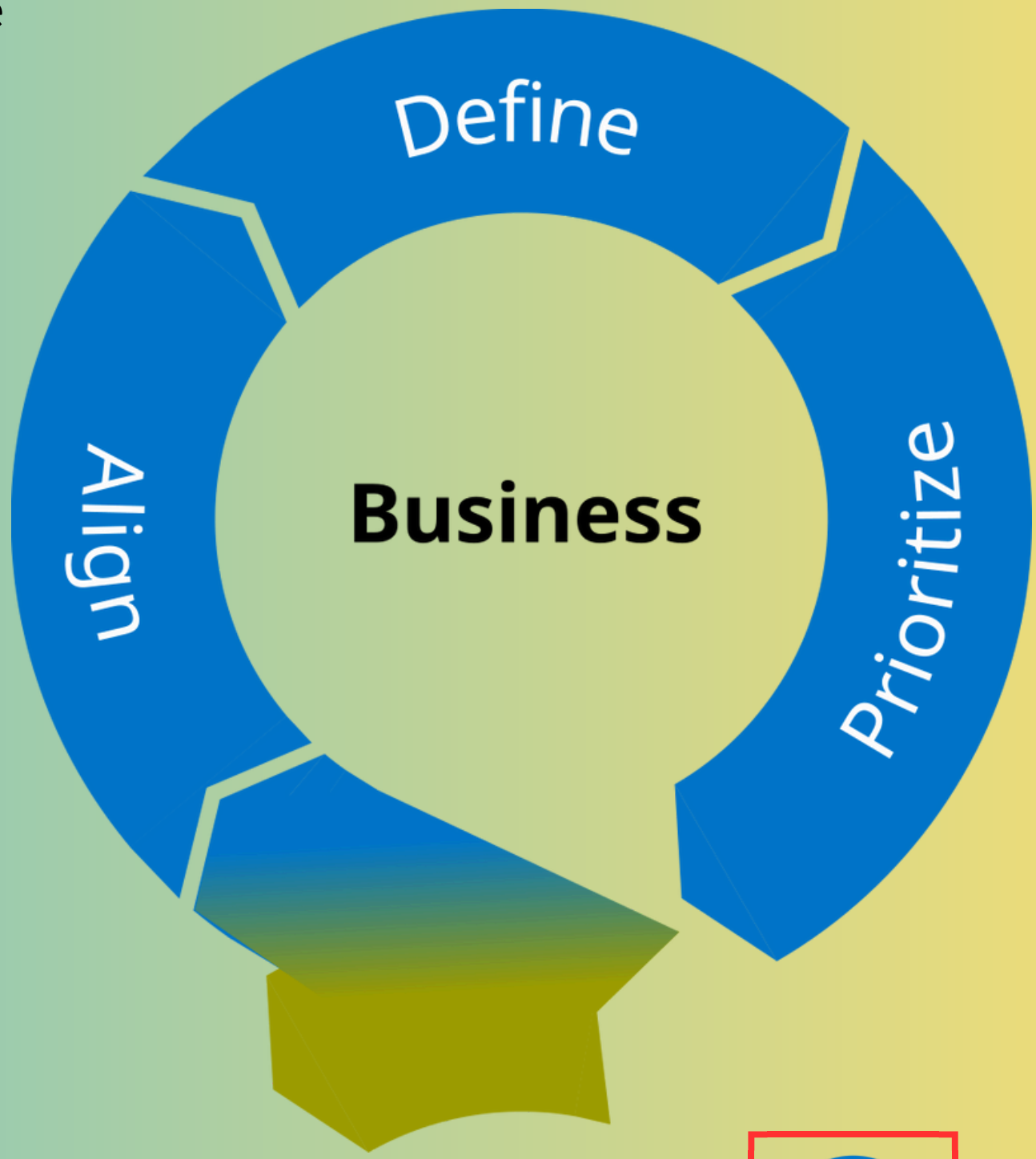
Business: Define Direction, Not Solutions

Provide value intent and priorities that guide development without locking in assumptions too early.

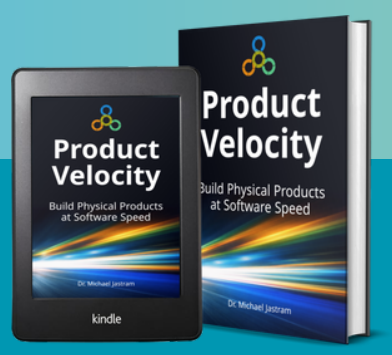
- **Define:** Articulate goals, value hypotheses, constraints, and success criteria. Avoid exhaustive upfront specification.

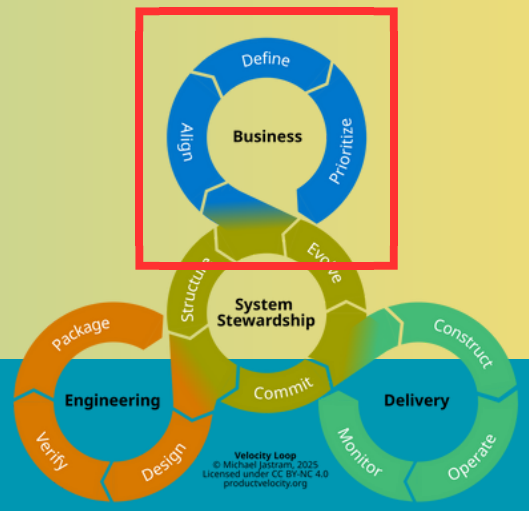
- **Align:** Adjust intent based on real-world feedback and system constraints. Keep definition viable, not complete.

- **Prioritize:** Set relative importance and sequencing. Enable decentralized technical decisions instead of acting as an approval gate.



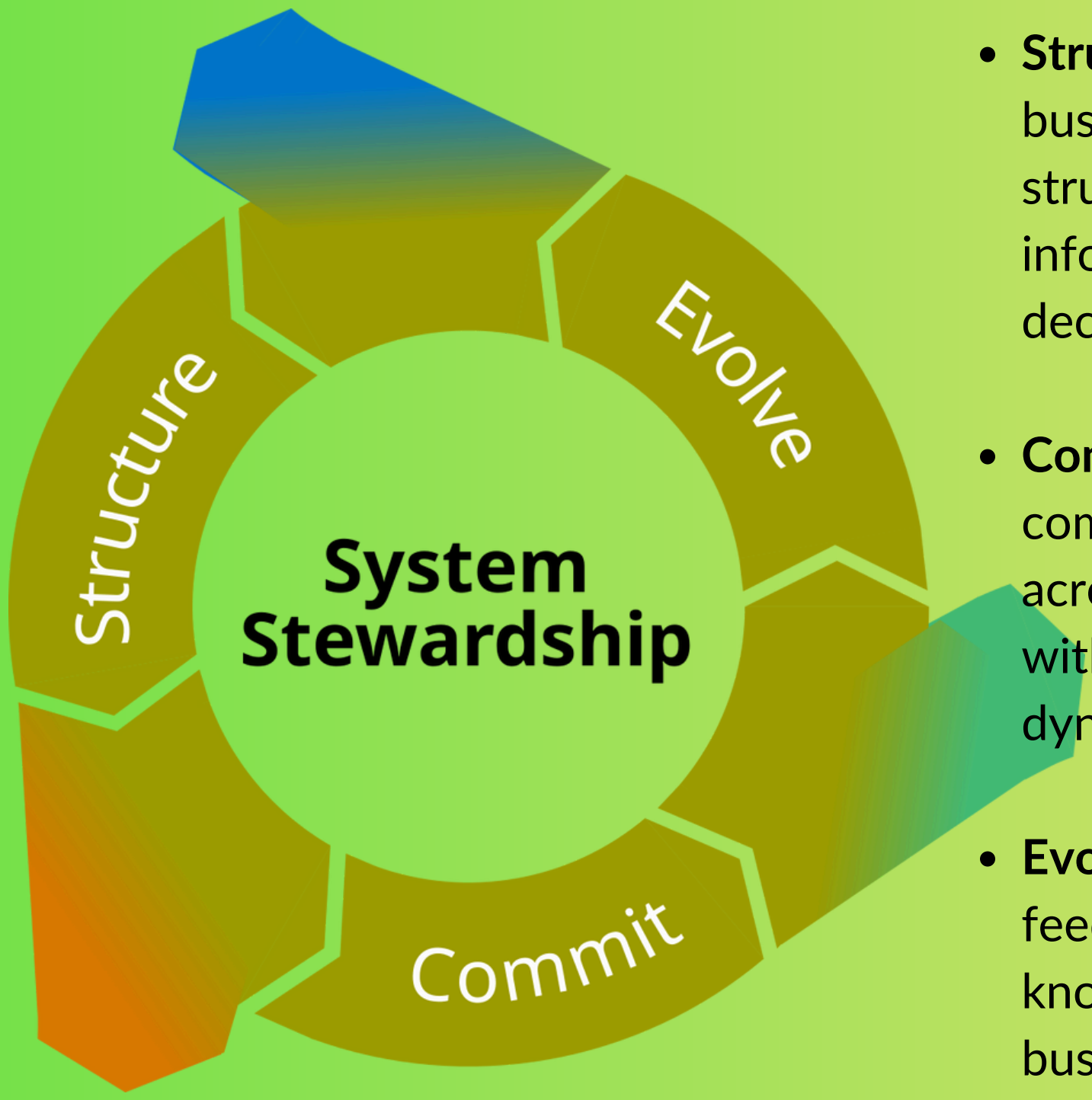
Key Output:
Intent and priorities that can evolve





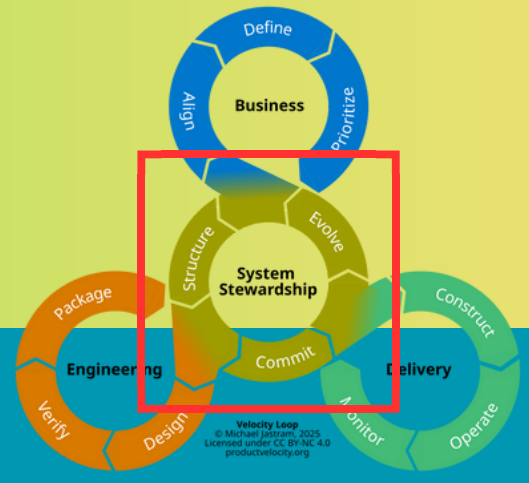
System Stewardship: Decision and Structure

Maintain architectural coherence, enable flow, and preserve knowledge as a single source of truth.



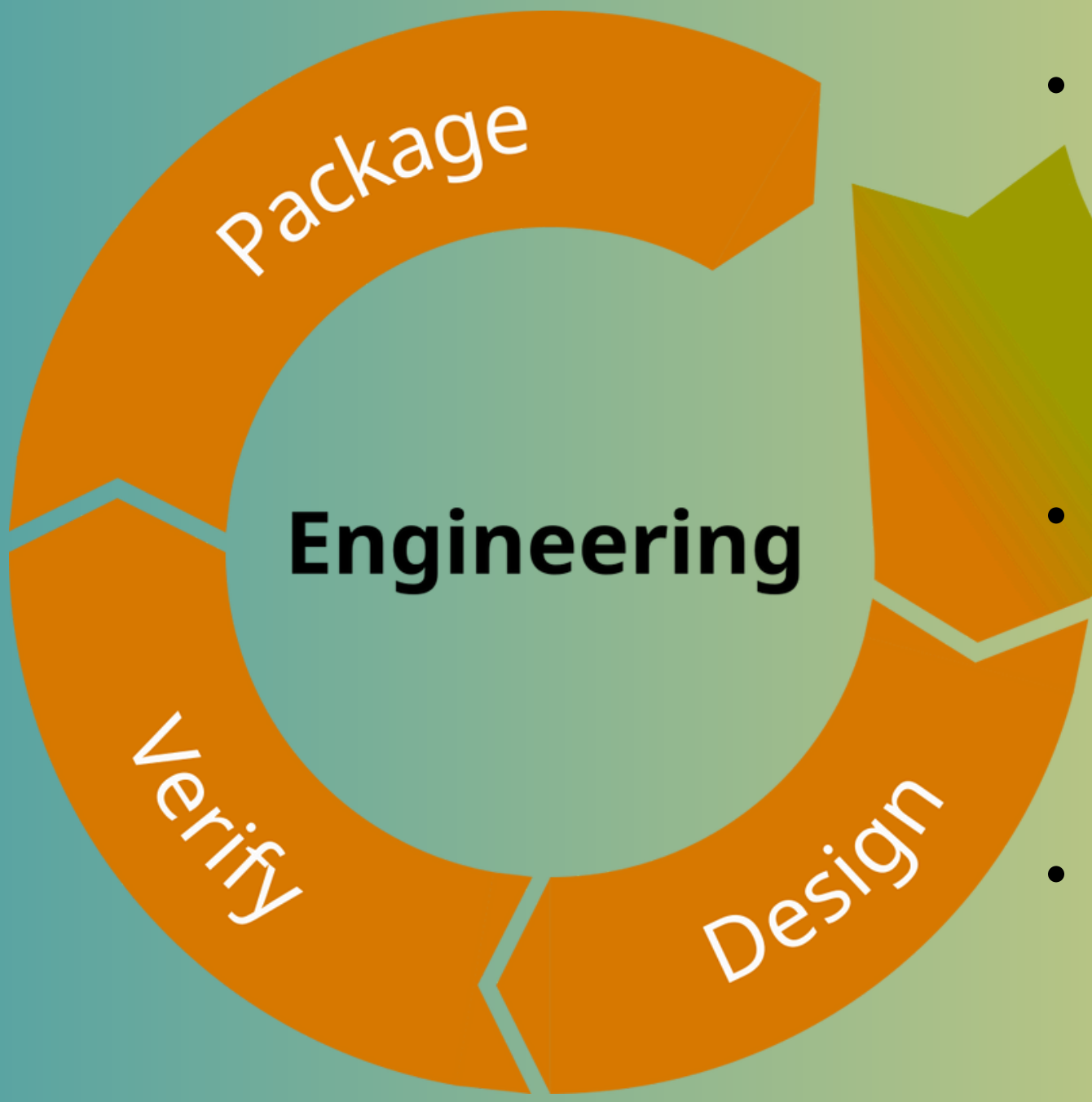
- **Structure:** Transform raw business input into structured, actionable system information. Decompose and decouple deliberately.
- **Commit:** Own system-level commitments and trade-offs across hardware and software with different change dynamics.
- **Evolve:** Convert delivery feedback into architectural knowledge and feed it back to business decisions.

Key Output:
Explicit architecture, preserved decisions, accumulated knowledge



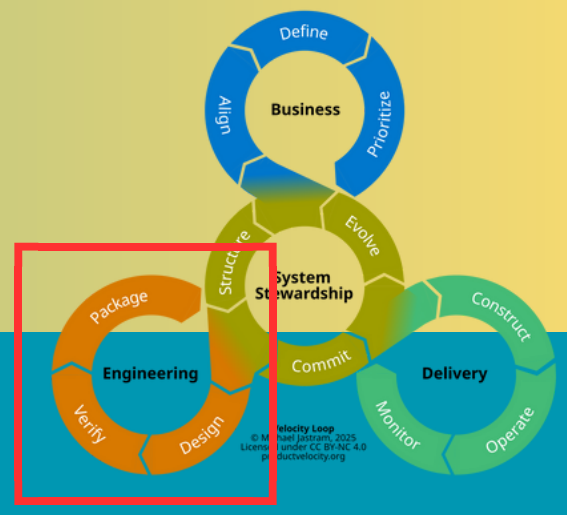
Engineering: Design for Flow and Feedback

Create buildable, verifiable system elements while optimizing end-to-end flow rather than local efficiency.



- **Design:** Implement solutions within defined interfaces and constraints. Favor black-box clarity over detailed prescriptions.
- **Verify:** Shift verification left using automation and simulation. Reduce late discovery and rework.
- **Package:** Ensure downstream readiness. Stabilize interfaces to enable parallel work and continuous integration.

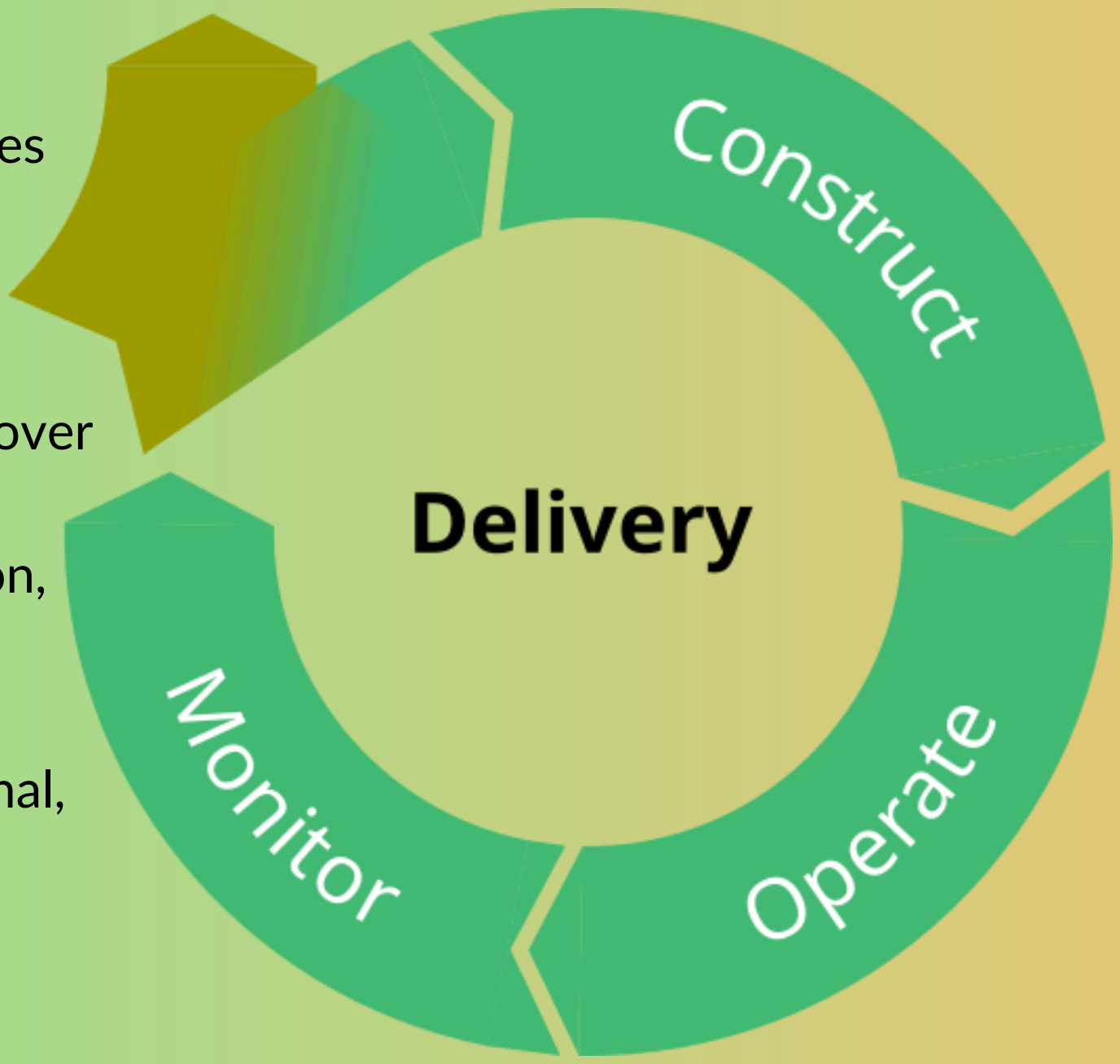
Key Output:
Executable artefacts ready for integration and delivery.



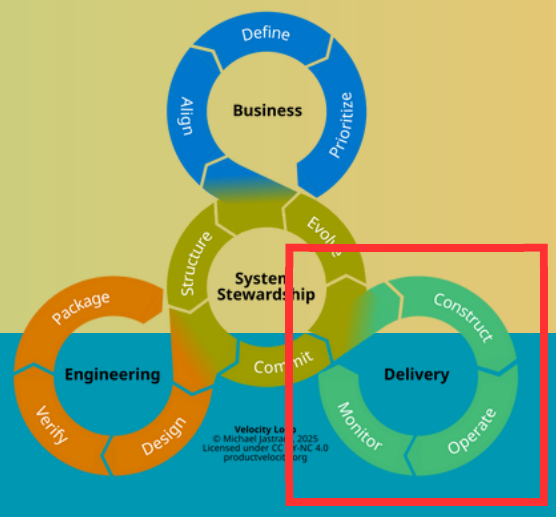
Delivery: Reality as the Final Judge

Turn designs into operational systems and generate feedback from real use.

- **Construct:** Build, manufacture, deploy. Explicitly handle differences between prototypes and series production.
- **Operate:** Run the system over time. Design for updates, maintenance, and evolution, not just launch.
- **Monitor:** Collect operational, usage, and quality data. Provide signals, not anecdotes.



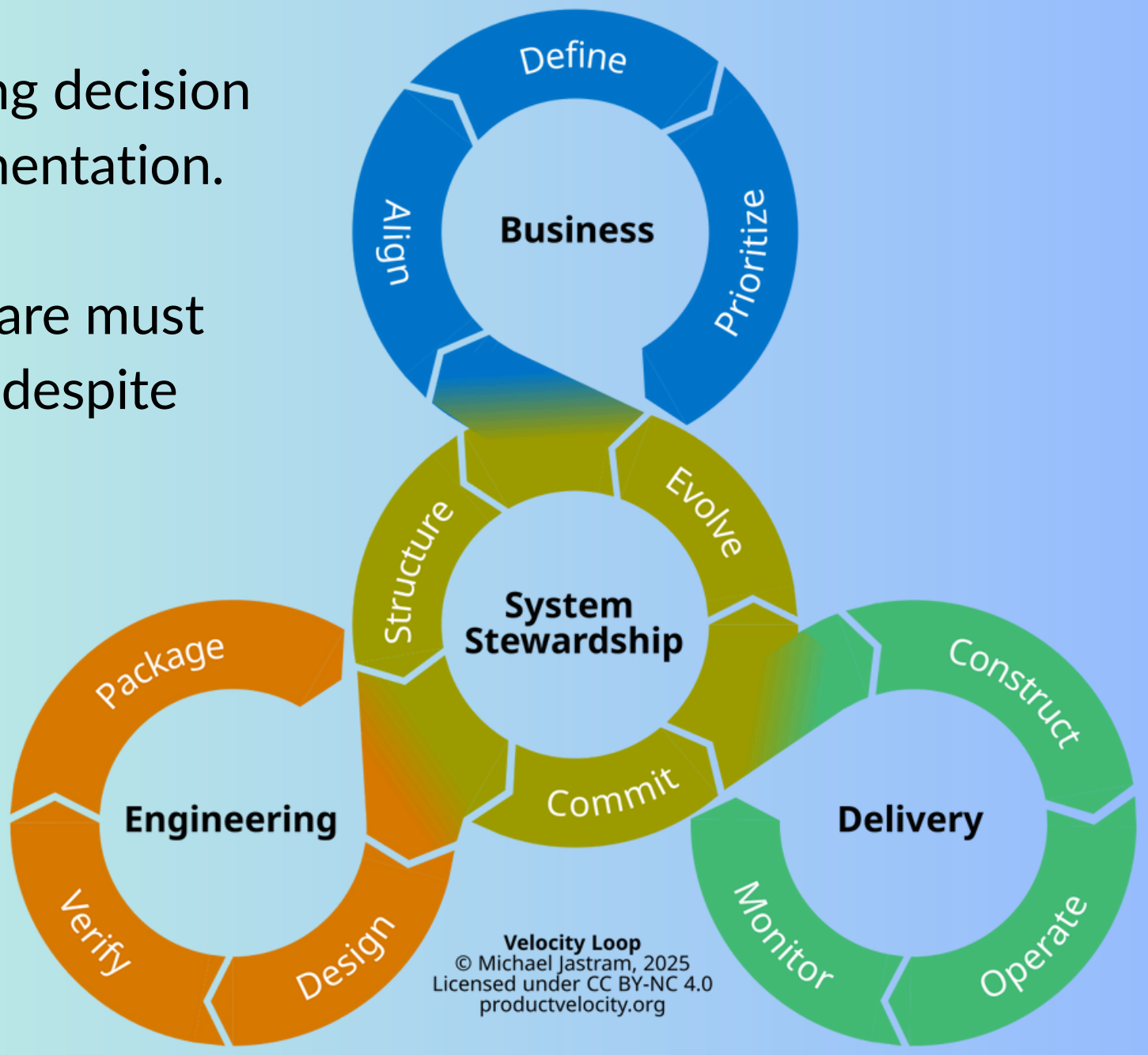
Key Output:
Instances and production feedback

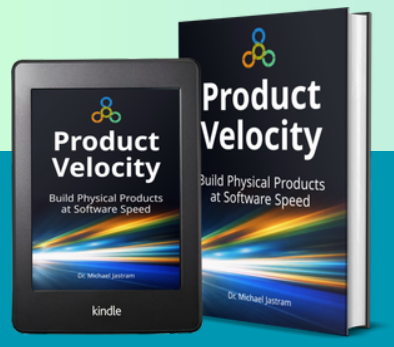


Why the Velocity Loop Matters

Speed comes from closed loops, not faster execution inside silos.

- Local optimization breaks system-level flow.
- Feedback that is collected but not absorbed is waste.
- Architecture is a living decision structure, not documentation.
- Hardware and software must be evolved together despite different rhythms.
- Sustainable speed requires explicit coordination, not heavier governance.





www.productvelocity.org

Book by Michael Jastram