

Planning and Leading Transformation to SDSE

A Practical Roadmap for Building Simulation-Driven Software Engineering
Capability

The central proposition

SDSE transformation is not a modeling-tool rollout. It is a change in how an organization defines intended behavior, resolves uncertainty, makes consequential decisions, and preserves evidence across the software lifecycle.

By Marc Hornbeek, a.k.a. DevOps_the_Gray

Originator of Simulation-Driven Software Engineering

August 2026

Executive Summary

Software organizations have invested heavily in Agile, DevOps, cloud platforms, automated testing, security pipelines, observability, and artificial intelligence. These capabilities have shortened delivery cycles and increased the amount of change an organization can produce. Yet many consequential failures still begin before code: stakeholders carry different assumptions about how the system should behave, important interactions remain implicit, failure and recovery are underspecified, and teams learn that the proposed behavior is unsafe or infeasible only after architecture and implementation commitments have hardened.

Simulation-Driven Software Engineering (SDSE) addresses this behavioral evidence gap. A team begins with intent and a Candidate Specification, represents consequential behavior in an executable model, simulates normal and challenging scenarios, examines traces and outcomes, revises flawed assumptions, and approves a versioned Validated Behavioral Baseline. Behavior Simulation Notation (BSN) and the BSN Workbench provide the notation and working environment. The baseline remains connected to design, code, tests, delivery controls, deployment, telemetry, incidents, and operational learning.

Adopting SDSE is therefore not simply a purchase, installation, or specialist training exercise. It changes the evidence used to make decisions, the responsibilities carried by product and engineering roles, the timing of learning, and the connection between pre-implementation intent and production reality. Leaders must create a credible reason for change, select bounded pilots, develop skills through real work, establish decision rights, connect the model with existing delivery systems, and measure whether earlier evidence improves outcomes.

This paper recommends a staged transformation: align and assess, prepare, prove, integrate, scale, and optimize. The first objective is one consequential decision improved by one bounded pilot. Scale should follow demonstrated value, not precede it. Reusable patterns, communities of practice, distributed Behavior Model Owners, lifecycle integrations, and proportional governance can then turn a successful experiment into an enterprise capability.

Core conclusion

The transformation succeeds when SDSE changes worthwhile decisions and preserves the resulting behavioral promise through delivery and operations. The objective is not more models. It is earlier learning, better commitments, safer behavior, and a continuously improving engineering system.

1. Lead with business and engineering consequences, not with enthusiasm for simulation technology.
2. Select pilots with meaningful uncertainty, measurable late-discovery cost, and accountable decision owners.

3. Model only enough behavior to change the selected decision.
4. Treat counterexamples as valuable learning rather than team failure.
5. Keep approval of intent, delegated authority, material change, and residual risk under accountable human control.
6. Scale through evidence, reusable assets, and distributed capability rather than a central modeling gate.

Contents

Executive Summary

- 1. The Transformation Imperative**
- 2. Understanding the Target State**
- 3. Principles for Leading SDSE Transformation**
- 4. Assessing Organizational Readiness**
- 5. Establishing the Transformation Vision and Business Case**
- 6. Building the Transformation Coalition**
- 7. Selecting the Right Initial Use Cases**
- 8. Designing and Executing the Pilot**
- 9. Integrating SDSE into the Software Lifecycle**
- 10. Integrating SDSE with Agile, DevOps, Governance, and AI**
- 11. Developing People and Capabilities**
- 12. Tooling and Architecture Strategy**
- 13. Governance and Decision Rights**
- 14. Measuring Transformation Progress and Value**
- 15. Scaling SDSE Across the Organization**
- 16. SDSE Transformation Maturity Model**
- 17. Transformation Roadmap**
- 18. Risks, Resistance, and Failure Modes**
- 19. Leadership Playbook**
- 20. The First 90 Days**
- 21. The Integrated SDSE Transformation System**

Conclusion

Appendices, References, and Author

1. The Transformation Imperative

Software engineering repeatedly advances by moving useful evidence earlier. Structured requirements made intent more explicit. Prototyping exposed usability concerns. Automated testing moved verification closer to development. Agile shortened the distance between implementation and stakeholder feedback. DevOps connected development with deployment and operations. SDSE advances the same trajectory by reducing the distance between behavioral intent and executable evidence.

1.1 The cost of discovering behavior late

Late behavioral discovery is expensive because misunderstanding propagates. An ambiguous promise influences estimates, architecture, interfaces, code, tests, security controls, deployment plans, operating procedures, customer expectations, and contractual commitments. A defect detected in code may require a local correction. A flawed behavioral assumption discovered after release can require redesign across multiple teams while the organization manages operational disruption and loss of trust.

The DoD Digital Engineering Strategy argues for authoritative digital models and a connected environment across conception, development, testing, fielding, and sustainment [1]. Its workforce plan emphasizes that new methods and tools affect requirements, engineering, test, cost, acquisition, and sustainment communities, not engineers alone [2]. SDSE applies a comparable transformation principle to software behavior: the model becomes useful only when it changes the system of work around it.

1.2 Why urgency is increasing

7. Distributed systems produce behavior through interactions that no single component owner fully controls.
8. Cloud, platform, and continuous-delivery practices increase the rate and reach of change.
9. Security, privacy, safety, reliability, and regulatory obligations depend on end-to-end behavior.
10. AI can generate designs, code, tests, configuration, and operational actions faster than people can review them manually.
11. Autonomous and agentic systems require explicit authority, escalation, stop, recovery, and human-control behavior.
12. Customers experience the combined behavior of the service, not the correctness of its individual artifacts.

1.3 The evidence gap

Dimension	Common condition	SDSE target condition
Intent	Discussed in meetings, documents, tickets, and policies	A shared, reviewable behavioral promise
Interaction	Often fragmented across teams and diagrams	Executable participants, states, interfaces, and sequences
Risk	Frequently addressed after design or code exists	Normal, boundary, failure, misuse, and recovery scenarios before commitment
Decision	Based on confidence, authority, and schedules	Based on explicit evidence thresholds and recorded assumptions
Lifecycle	Requirements drift away from implementation and operations	A versioned baseline linked to code, tests, controls, and telemetry
Learning	Incidents correct production without updating intent	Operational evidence revises and resimulates the living model

Table 1. SDSE transformation closes a lifecycle-wide behavioral evidence gap.

2. Understanding the Target State

Leaders cannot direct a transformation until they can describe the future operating condition in observable terms. The target is not an organization that produces a prescribed number of models. It is an organization that makes selected consequential behavior executable early, uses evidence to improve decisions, and keeps the approved behavioral promise connected to the delivered and operated system.

2.1 The SDSE operating sequence

1. **Establish intent.** Identify stakeholders, desired outcomes, obligations, unacceptable outcomes, constraints, and the decision that requires evidence.
2. **Create a Candidate Specification.** State the proposed behavior clearly enough to expose assumptions and open questions without pretending the specification is already complete.
3. **Build the behavior model.** Represent relevant participants, boundaries, interfaces, conditions, state, logic, timing, interactions, and expectations using BSN.
4. **Simulate by risk.** Run normal, boundary, degraded, failure, misuse, recovery, and human-intervention scenarios proportionate to consequence and uncertainty.
5. **Review the evidence.** Inspect traces, failed expectations, contradictions, sensitivities, omissions, and model limitations.

6. **Approve a baseline.** Accept, revise, reject, restrict, defer, or observe the proposed behavior; record the version, evidence, limits, authority, and residual risk.
7. **Engineer and compare.** Use the baseline to guide architecture, implementation, testing, delivery controls, and production conformance.
8. **Learn and revise.** Feed operational difference, incidents, usage, and environmental change back into a new candidate version.

2.2 Behavior Simulation Notation and the BSN Workbench

BSN provides a behavior-focused notation for describing boxes, interfaces, operating conditions, logic, networks, and scenarios. The BSN Workbench supports model authoring, simulation, traces, expectations, comparison, version continuity, and the evidence needed for review. The tool is important, but its purpose is to enable a disciplined decision system. A technically elegant model that does not inform an accountable decision is transformation activity without transformation value.

2.3 Observable characteristics of the target state

9. Consequential backlog items identify the behavioral question and evidence needed for a decision.
10. Candidate Specifications expose uncertainty instead of hiding it behind premature precision.
11. Simulation results are reviewed by product, engineering, quality, security, reliability, and operations as appropriate.
12. Validated Behavioral Baselines are versioned and linked to implementation and operational evidence.
13. Material change triggers review and proportional resimulation.
14. Teams can explain model scope, fidelity, assumptions, exclusions, and residual risk.
15. Production reality is allowed to contradict the model, and the model is then corrected.

3. Principles for Leading SDSE Transformation

Transformation principles are decision rules for moments when the detailed roadmap does not provide an answer. They protect the purpose of SDSE while allowing practices to adapt to different products, risks, delivery methods, and organizational structures.

1. **Lead with decisions, not models.** Name the decision before defining model scope. Evidence must be capable of changing that decision.
2. **Apply proportionality.** Increase scope and fidelity with consequence, uncertainty, interaction complexity, autonomy, and cost of late discovery.

3. **Begin with candidates.** Simulation is a way to discover what the specification is missing; completeness is an outcome of learning, not a prerequisite for starting.
4. **Reward counterexamples.** A scenario that disproves an assumption creates value while change is still comparatively inexpensive.
5. **Preserve human accountability.** AI and tools may propose or execute; accountable people authorize intent, limits, material change, and residual risk.
6. **Keep the model alive.** A baseline is a versioned reference for controlled change, not a frozen document.
7. **Integrate into existing work.** SDSE should strengthen product, Agile, DevOps, security, reliability, and governance practices rather than create a parallel bureaucracy.
8. **Scale through pull.** Teams adopt and reuse practices because they improve decisions and outcomes, not because a central function demands artifacts.

Transformation test

If the adoption program can report many models but cannot identify decisions changed, rework avoided, risks exposed, or outcomes improved, it is measuring activity rather than transformation.

4. Assessing Organizational Readiness

Readiness is not a pass-or-fail judgment about whether an organization deserves to begin. It identifies the conditions a pilot must create or compensate for. A team can start with modest tooling maturity when it has a real decision, a cooperative cross-functional group, access to stakeholders, and permission to revise assumptions. Conversely, sophisticated technology cannot rescue a pilot with no decision owner or no willingness to act on evidence.

Readiness dimension	Evidence of readiness	Warning sign
Business	Consequential outcomes and late-discovery costs are understood	No sponsor or no business consequence is named
Leadership	Sponsor protects learning and permits evidence to change commitments	Pilot is expected to validate a predetermined answer
Product	Behavioral promise and decision owner can be identified	Work is framed only as features or solution tasks
Engineering	Architecture, implementation, quality, security, and operations can participate	Modeling is delegated to an isolated specialist
Culture	Teams can expose uncertainty and challenge assumptions safely	Contradiction is treated as delay, resistance, or failure
Information	Relevant policies, interfaces, incidents, and operating evidence are accessible	Critical context is unavailable or untrusted

Readiness dimension	Evidence of readiness	Warning sign
Tooling	A repository and BSN Workbench environment can be established	Access, security, or ownership blocks practical use
Measurement	A baseline and comparison period can be defined	Success is limited to training completion or model count

Table 2. Readiness assessment identifies enabling conditions and pilot risks.

4.1 Readiness assessment outputs

9. A concise current-state map of how intent, decisions, assurance, and operational learning currently flow.
10. A list of behavioral evidence gaps with representative consequences.
11. A stakeholder and capability map covering sponsorship, decision authority, participants, and affected roles.
12. Candidate pilots ranked by value, feasibility, learning potential, and measurable evidence.
13. Constraints involving data, security, intellectual property, tools, integration, schedules, or suppliers.
14. An initial change-impact view describing what each role must do differently.

5. Establishing the Transformation Vision and Business Case

A credible vision connects SDSE with a problem leaders and teams already recognize. “Adopt behavior simulation” describes an activity. “Reduce redesign caused by cross-service behavioral ambiguity” describes an outcome. The transformation narrative should explain the present consequence, the future capability, why the change is practical, and how progress will be measured.

5.1 Frame the problem in operational language

15. Requirements clarification consumes substantial development time after work begins.
16. Integration reveals incompatible assumptions across teams or suppliers.
17. Planning is unstable because states, dependencies, failure, or authority were not understood.
18. Security and reliability scenarios are discovered after architecture or implementation is difficult to change.
19. AI-generated artifacts move faster than reviewers can establish whether the intended behavior is acceptable.

20. Production incidents reveal that tests proved components while the end-to-end promise remained wrong.

5.2 Define a measurable value hypothesis

The business case should distinguish investment, leading evidence, delivery outcomes, and business consequences. The pilot does not need to promise a precise enterprise return before data exists. It does need a falsifiable hypothesis: if SDSE improves the selected decision, which observable measures should move, over what period, and compared with what baseline?

Value layer	Transformation question
Investment	Training, facilitation, modeling, Workbench setup, review, integration, and maintenance
Capability	Teams express behavior, simulate scenarios, interpret evidence, and govern baselines
Leading evidence	Questions, contradictions, dependencies, unsafe paths, and recovery gaps found earlier
Delivery outcome	Less clarification during build, lower rework, better forecast stability, fewer behavioral escapes
Business outcome	Faster value realization, lower failure cost, stronger trust, safer autonomy, better compliance evidence

Table 3. The business case connects SDSE activity to delivery and business outcomes.

5.3 Create the transformation charter

Charter element	Required answer
Problem and consequence	What recurring or material outcome justifies action?
Target capability	What will teams be able to do that they cannot do reliably today?
Scope and exclusions	Which product, decision, scenario, or value stream is included and excluded?
Sponsor and authority	Who can allocate resources and act when evidence changes commitments?
Pilot and participants	Which real work will demonstrate the capability, and who must participate?
Measures and baseline	What will be compared, over what period, and from which evidence systems?
Decision gates	What results lead to stop, revise, repeat, integrate, or scale?

Table 4. A concise charter protects the pilot from vague ambition and premature expansion.

6. Building the Transformation Coalition

SDSE spans intent, engineering, assurance, delivery, and operation. It cannot be owned successfully by one function working after decisions are made. Kotter's change framework emphasizes urgency, a

guiding coalition, a strategic vision, removal of barriers, and visible short-term wins [3]. For SDSE, the coalition must combine authority with direct knowledge of the product and the system of work.

Coalition role	Primary transformation accountability
Executive sponsor	Business outcome, authority, resources, and protection for evidence-based change
Transformation leader	Roadmap, dependencies, risks, communication, measures, and cross-functional coordination
Product leader	Outcome, behavioral promise, priority, stakeholders, and evidence needed for product decisions
Engineering leader	Architecture, implementation, technical integration, staffing, and conformance strategy
Quality / security / reliability	Challenging scenarios, unacceptable outcomes, controls, recovery, and independent evidence
Operations representative	Operating conditions, telemetry, incident evidence, safe deployment, and learning triggers
Behavior Model Owner	Model continuity, version integrity, assumptions, scenarios, decisions, links, and lifecycle updates
SDSE coach or champion	Facilitation, training, patterns, review quality, proportionality, and capability transfer
Decision authority	Evidence sufficiency, accepted limits, approval, delegated authority, and residual risk

Table 5. The coalition joins change leadership with accountable product and engineering authority.

6.1 Keep sponsorship active

Sponsorship is not a signature on the charter. The sponsor must communicate why the behavioral evidence gap matters, protect the pilot when simulation exposes uncomfortable assumptions, remove cross-functional barriers, and use the resulting evidence in a real decision. If leaders request learning but punish changes to scope or schedule, teams will produce reassuring models rather than truthful ones.

6.2 Keep ownership distributed

A central enablement group can supply training, coaching, patterns, standards, and integrations. It should not become the only group permitted to model behavior or approve product decisions. Product and engineering teams remain accountable for what they build and operate. The center enables capability; it does not inherit everyone else's responsibility.

7. Selecting the Right Initial Use Cases

The first pilot is an economic and cultural experiment. It should be important enough to demonstrate value, bounded enough to complete within a useful time, and measurable enough to distinguish improved decisions from persuasive storytelling. The ideal pilot contains real behavioral uncertainty and a decision that has not already been made.

Criterion	Assessment question	Score
Business consequence	Would a late misunderstanding materially affect value, cost, safety, trust, or obligation?	0-5
Behavioral uncertainty	Are important interactions, states, timing, authority, failure, or recovery unresolved?	0-5
Late-discovery cost	Would correction after architecture, code, integration, or release be expensive?	0-5
Boundary and feasibility	Can a meaningful model and scenario set be completed within the pilot period?	0-5
Decision leverage	Can simulation evidence genuinely change scope, design, controls, release, or observation?	0-5
Stakeholder access	Can the required product, domain, engineering, assurance, and operations participants engage?	0-5
Measurement quality	Can the team establish a baseline and collect comparable evidence?	0-5
Reuse potential	Will patterns, scenarios, skills, or integrations benefit later work?	0-5

Table 6. Pilot selection scorecard. A high total does not override a critical feasibility or authority gap.

7.1 Strong pilot patterns

- 21. Cross-service workflows with incompatible assumptions or expensive integration history.
- 22. Safety-, security-, privacy-, reliability-, or policy-sensitive behavior.
- 23. Failure, retry, timeout, rollback, recovery, and degraded-mode behavior.
- 24. AI-agent permissions, escalation, delegation, and human override.
- 25. State, sequence, concurrency, or timing behavior that is difficult to reason about informally.
- 26. Multi-team or supplier boundaries where contracts describe interfaces but not complete interaction behavior.

7.2 Pilot traps

- 27. Selecting trivial work because it is politically safe, then concluding that SDSE has little value.
- 28. Selecting a program so large that no decision can be completed or measured.
- 29. Starting after architecture and commitments are fixed, reducing simulation to documentation.
- 30. Choosing a pilot with no stakeholder empowered to revise the proposed behavior.
- 31. Attempting to automate every integration before proving that the evidence matters.

8. Designing and Executing the Pilot

A disciplined pilot is small enough to learn quickly but complete enough to test the full value chain from intent to decision. It should create a useful behavioral baseline, connect that baseline to real delivery work, and compare outcomes with a prior or parallel baseline.

1. **Establish the baseline.** Record clarification effort, decision latency, estimate change, rework, behavioral defects, integration issues, and relevant outcome evidence before the pilot.
2. **Define the decision.** State what the team must decide and what evidence can cause it to accept, revise, reject, restrict, defer, or observe.
3. **Write the Candidate Specification.** Capture participants, conditions, state, constraints, interactions, expectations, failure, recovery, assumptions, and open questions.
4. **Set proportional scope.** Select only behavior necessary to reduce the uncertainty around the decision.
5. **Build the BSN model.** Create a reviewable executable representation in the BSN Workbench and identify model fidelity and exclusions.
6. **Design scenarios by consequence.** Prioritize normal, boundary, failure, misuse, degraded, recovery, timing, concurrency, and authority scenarios.
7. **Simulate and review.** Bring the required disciplines together to interpret traces and counterexamples rather than merely inspect a demonstration.
8. **Decide and baseline.** Record the accepted behavior, evidence, limits, assumptions, approvals, and residual risk.
9. **Connect the delivery work.** Link the baseline to backlog, architecture, code, tests, controls, release criteria, and telemetry.
10. **Measure and learn.** Compare pilot outcomes with the baseline, document confounding changes, and decide whether to stop, adjust, repeat, integrate, or scale.

Pilot completion rule

The pilot is not complete when the model runs. It is complete when accountable people use its evidence in a decision, connect the result to delivery, and evaluate the outcome.

9. Integrating SDSE into the Software Lifecycle

SDSE creates its greatest value when the Validated Behavioral Baseline becomes a shared reference rather than a pre-development artifact. Each lifecycle discipline contributes different evidence. The model guides work downstream, while implementation and operational reality challenge the model upstream.

Lifecycle activity	SDSE integration
Product discovery	Behavioral hypotheses, stakeholder outcomes, unacceptable outcomes, and evidence needed to choose among alternatives
Backlog refinement	Participants, conditions, state, dependencies, constraints, scenarios, assumptions, and open questions

Lifecycle activity	SDSE integration
Planning	Better scope, dependency visibility, risk exposure, evidence work, and forecast confidence
Architecture and design	Alternative evaluation against interaction, scale, failure, security, recovery, and operational behavior
Development	A versioned behavioral reference for AI-assisted and human implementation decisions
Testing and assurance	Scenario-derived expectations and comparison between modeled and implemented behavior
Deployment and release	Behavior-linked controls, evidence thresholds, rollback, recovery, and accepted residual risk
Operations	Telemetry and incident evidence compared with the intended baseline and operating assumptions
Learning	Material divergence creates a new candidate version for review, resimulation, and controlled adaptation

Table 7. The living model carries one behavioral promise through the lifecycle.

9.1 Definition of Ready

For selected consequential work, readiness may require a clear outcome and promise, visible participants and state, known constraints, owned material questions, proportional scenarios, and an identified decision authority. Readiness does not mean the model is complete. It means the team understands what it must learn and has a credible way to obtain evidence.

9.2 Definition of Done

Completion may require implementation evidence for the agreed scenarios, conformance or approved deviation from the baseline, operating controls and telemetry, recorded residual uncertainty, and a clear learning trigger. The production system remains the product; the model is the reference that makes comparison and adaptation coherent.

10. Integrating SDSE with Agile, DevOps, Governance, and AI

10.1 Agile

Agile values remain essential. SDSE adds executable learning before substantial implementation without requiring complete up-front specification. Candidate models remain small, revisable, and tied to current decisions. Backlog refinement becomes more behaviorally explicit; reviews can demonstrate both the increment and its relationship to the approved promise; retrospectives can improve the model as well as the process. The Agile Manifesto's emphasis on working software, collaboration, and response to change remains intact [9].

10.2 DevOps and continuous delivery

DevOps provides the flow, automation, feedback, and operational collaboration needed to carry the behavioral baseline into delivery. Model evidence can inform test selection, deployment controls, approval policies, observability, and incident learning. A green pipeline remains necessary but is interpreted correctly: it proves configured controls passed, not that unmodeled behavior is safe.

10.3 Security and governance

NIST's Secure Software Development Framework is designed to integrate secure practices into different software development lifecycle implementations [6]. SDSE complements that approach by making selected security-relevant business and system behavior executable. Governance becomes an outcome of integrated intent, evidence, authority, controls, and traceability rather than a separate approval ceremony.

10.4 AI-native engineering

DORA's 2024 research reported widespread AI use alongside gains in individual productivity and cautioned that improved development work does not automatically improve delivery performance without sound delivery fundamentals [5]. SDSE supplies an additional foundation: AI can assist in proposing specifications, models, scenarios, code, and tests, but accountable humans must approve the behavioral promise, permission boundaries, material change, and residual risk. NIST's AI RMF similarly frames governance, mapping, measurement, and management as connected risk functions across the AI lifecycle [7].

11. Developing People and Capabilities

Transformation happens when people perform new behaviors reliably, not when a tool is available. Prosci's methodology distinguishes organizational change from the individual journey represented by awareness, desire, knowledge, ability, and reinforcement [4]. SDSE leaders should therefore plan both the engineering rollout and the role-specific adoption path.

Audience	Required capability	Development method
Executives and sponsors	Behavioral evidence gap, value case, proportionality, sponsorship, decision use	Orientation and sponsor coaching
Product and program leaders	Behavioral promise, candidate specification, evidence thresholds, prioritization	Workshop plus facilitated pilot work
Engineers and architects	BSN modeling, scenario challenge, traces, alternatives, implementation conformance	Practitioner training and paired modeling

Audience	Required capability	Development method
Quality, security, and reliability	Adverse scenarios, unacceptable outcomes, independent challenge, recovery evidence	Domain labs using real risks
Operations	Operating conditions, telemetry mapping, drift, incident-to-model feedback	Lifecycle integration exercises
Behavior Model Owners	Version integrity, scope, evidence, assumptions, links, review, maintenance	Advanced training and supervised ownership
Coaches and champions	Facilitation, proportionality, patterns, adoption barriers, measurement	Train-the-coach and community practice

Table 8. Capability development should be role-specific and grounded in real work.

11.1 Learn through consequential work

Classroom examples are useful for syntax and concepts, but transformation competence is built when participants model their own domain, negotiate conflicting expectations, interpret evidence, and make a decision. Coaching should be strongest during the first models and gradually reduce as teams demonstrate independent judgment.

11.2 Establish a community of practice

- 11.Share reusable model elements, scenarios, domain patterns, and integration techniques.
- 12.Review difficult questions about scope, fidelity, model risk, and evidence sufficiency.
- 13.Publish lessons from failed assumptions without turning them into blame narratives.
- 14.Maintain practical guidance and exemplars instead of a large static methodology manual.
- 15.Identify recurring friction that requires leadership, architecture, tooling, or policy intervention.

12. Tooling and Architecture Strategy

The BSN Workbench is the execution environment for the living behavior model, but the transformation architecture includes every system that creates, consumes, or challenges behavioral evidence. Tool strategy should follow value: prove the workflow with light connections, then automate integrations that reduce material friction or strengthen traceability.

12.1 Establish the Workbench operating model

- 16.Define installation, access, security, backup, and environment responsibilities.
- 17.Create naming, repository, ownership, versioning, and baseline conventions.
- 18.Separate candidate work from approved baselines while preserving history.

19. Define how scenarios, expectations, traces, decisions, and limitations are retained.
20. Protect sensitive business behavior, security scenarios, data assumptions, and intellectual property.
21. Identify archival, audit, and supplier-exchange requirements where relevant.

12.2 Integrate incrementally

System	High-value integration
Product and backlog	Link decision, model version, scenarios, and open questions to selected work items
Architecture repository	Trace structural decisions and alternatives to behavioral evidence
Source control	Associate implementation changes with the applicable behavioral baseline
Test management	Relate simulation scenarios and expectations to implementation evidence
CI/CD	Evaluate relevant conformance and release evidence without creating universal gates
Security and policy	Connect threat, misuse, authority, control, and evidence requirements
Observability and incidents	Compare production behavior with assumptions and trigger model review

Table 9. Integrations should preserve the behavioral evidence thread, not automate for its own sake.

Architecture rule

Do not delay the pilot while attempting to integrate every system. First prove that the evidence changes a worthwhile decision; then automate the connections that protect or accelerate that value.

13. Governance and Decision Rights

Governance should make authority, evidence, and residual risk visible without centralizing every decision. The level of review must be proportional to consequence and uncertainty. Routine behavior can remain within team authority. Higher-risk behavior may require independent challenge, executive approval, regulatory evidence, or restricted deployment.

Decision	Accountable authority	Minimum evidence
Define product intent and unacceptable outcomes	Product or business authority	Stakeholder need, obligation, policy, and consequence
Approve model scope and fidelity	Accountable product and engineering leaders	Decision relevance, risk, assumptions, and exclusions
Approve the Validated Behavioral Baseline	Named decision authority	Scenario evidence, failed expectations, limitations, and residual risk
Accept material behavioral change	Authority proportionate to consequence	Delta analysis, affected scenarios, resimulation, and approvals

Decision	Accountable authority	Minimum evidence
Authorize AI or agent autonomy	Human accountable for delegated authority	Permission, escalation, stop, recovery, and monitoring evidence
Authorize release	Existing release authority	Implementation evidence, deviation, controls, rollback, and operating protection
Require model revision	Product, engineering, assurance, or operations under policy	Incident, drift, changed assumption, regulation, environment, or outcome

Table 10. Decision rights keep authority close to the work while making consequential approval explicit.

13.1 Model risk

A model can be incomplete, incorrect, stale, or misleadingly precise. Every consequential baseline should state purpose, scope, fidelity, assumptions, exclusions, scenario coverage, validation method, and known divergence from production. Independent challenge is appropriate when consequence is high or when the model's authors share the same assumptions as the proposed solution.

13.2 Governance as an outcome

Good governance emerges when the delivery system integrates intent, evidence, controls, decision authority, traceability, and learning. It is not a separate pillar added after engineering. SDSE strengthens governance because it makes selected behavioral promises executable and reviewable before those promises are converted into production consequences.

14. Measuring Transformation Progress and Value

The measurement system must answer three questions: Are people adopting the capability? Is the capability improving engineering decisions and delivery? Are those improvements producing worthwhile product or business outcomes? Activity measures can help diagnose adoption, but they should not be mistaken for value.

Dimension	Possible measure	Evidence source
Adoption	Role participation; trained practitioners applying the practice; independently completed models	Training and Workbench records
Behavioral clarity	Material questions, contradictions, and missing expectations resolved before implementation	Refinement and decision records
Decision latency	Time from consequential question to accept, revise, reject, restrict, defer, or observe	Workflow timestamps
Planning	Estimate change after clarification; dependency discoveries; forecast stability	Planning history
Flow	Blocked work and time lost to behavioral clarification during delivery	Delivery system
Early discovery	Unsafe interactions, missing controls, and recovery gaps found before code or release	Simulation findings

Dimension	Possible measure	Evidence source
Rework	Behavior-related redesign, recoding, retesting, integration correction, and release delay	Change and time records
Quality and delivery	Escaped behavioral defects, change failures, rollback, and recovery effort	Test, pipeline, and incident systems
Learning	Time from operational evidence to model revision, resimulation, and approved response	Model and release history
Business outcome	Customer, financial, operational, safety, policy, or trust result tied to the validated promise	Product analytics and business records

Table 11. Balanced measurement connects adoption, engineering performance, and business outcomes.

14.1 Measurement safeguards

22. Establish the comparison baseline before the pilot begins.
23. Include training, facilitation, modeling, review, integration, maintenance, and delay in the cost.
24. Record other process, architecture, staffing, tooling, and market changes that could explain results.
25. Do not count every counterexample as a production incident that was certainly prevented.
26. Report unfavorable evidence and stop modeling that does not improve a worthwhile decision.
27. Never use model or scenario counts as individual performance targets.

15. Scaling SDSE Across the Organization

Scaling begins when a pilot produces a credible result and reusable learning. A mandate issued before teams experience value is likely to create compliance artifacts, specialist queues, and resistance. A better sequence is to make the work easier, demonstrate that it protects important decisions, and allow demand to grow around proven patterns.

15.1 Scale through reuse

28. Reusable BSN model elements for common domain participants and interfaces.
29. Scenario libraries for timeout, retry, partial failure, degraded mode, authority, misuse, and recovery.
30. Candidate Specification prompts and evidence review checklists.
31. Integration components linking models with backlog, code, tests, pipelines, and telemetry.
32. Proportionality policies that clarify which work benefits from which level of modeling and review.
33. Case examples showing the decision changed, the evidence used, the cost incurred, and the outcome observed.

15.2 Establish a center of enablement

The center of enablement should train, coach, curate assets, operate shared tooling, improve integrations, monitor adoption barriers, and support difficult cross-team problems. Its success is measured by independent team capability and reuse, not by the volume of work routed through the center.

15.3 Coordinate across value streams

Enterprise value often depends on behavior spanning products, platforms, suppliers, policies, and operations. Scaling therefore requires agreements about shared model elements, version compatibility, scenario ownership, cross-team evidence, and escalation. The organization should federate ownership while maintaining enough common structure to simulate the behavior customers and operators actually experience.

16. SDSE Transformation Maturity Model

Maturity describes organizational capability, not prestige. A team should advance only when a higher level solves a real problem. The objective is not to force every product into the same state, but to make required capability explicit and invest proportionately.

Level	Observable condition	Primary investment
1. Awareness	Leaders and teams understand the behavioral evidence gap; candidate use cases are identified	Orientation, readiness assessment, sponsor alignment
2. Experimentation	Bounded pilots use BSN and the Workbench to improve selected decisions	Facilitated pilots, baseline measures, documented lessons
3. Repeatable practice	Roles, workflows, templates, measures, and ownership are defined and reused	Trained practitioners, Behavior Model Owners, scenario patterns
4. Lifecycle integration	Baselines connect to product, architecture, code, tests, delivery, and operations	Tool integrations, governance, operational feedback
5. Adaptive enterprise capability	Behavioral evidence supports decisions across value streams and continuous change	Federated ownership, reuse, analytics, human-controlled AI assistance

Table 12. Maturity increases from awareness to an adaptive, lifecycle-wide capability.

16.1 Evidence for advancement

Advancement should be based on repeatable outcomes: more than one team can perform the practice, ownership survives personnel changes, models remain current through delivery, operational evidence triggers revision, and leaders use the evidence in decisions. Training completion alone does not establish maturity.

17. Transformation Roadmap

Phase	Primary work	Exit evidence
1. Align and assess	Sponsor, problem, readiness, candidate pilots, baseline hypotheses	Transformation charter and selected pilot
2. Prepare	Role training, Workbench setup, decision rights, repository, baseline measures	Pilot team ready to work
3. Prove	Candidate Specification, BSN model, scenarios, simulation, decision, delivery connection	Measured pilot result and lessons
4. Integrate	Lifecycle workflow, ownership, evidence links, governance, operational feedback	Repeatable practice within the value stream

Phase	Primary work	Exit evidence
5. Scale	Coaches, reusable assets, communities, federated ownership, targeted automation	Multiple teams applying SDSE independently
6. Optimize	Conformance analytics, model/production comparison, portfolio learning, AI assistance	Adaptive enterprise capability

Table 13. The roadmap scales demonstrated capability rather than unproven process.

17.1 Portfolio governance

At portfolio level, leaders should review value hypotheses, evidence quality, capability constraints, cross-team dependencies, reusable assets, integration investment, and the distribution of maturity required by product risk. The portfolio should fund learning and enablement as well as delivery. It should also stop work when simulation activity no longer changes decisions.

18. Risks, Resistance, and Failure Modes

Misconception	Failure created	Leadership safeguard
"SDSE is another tool rollout."	Teams learn syntax without changing decisions or lifecycle practice	Charter every pilot around a consequential problem and decision
"We must model everything."	Cost, delay, and resistance overwhelm value	Use consequence, uncertainty, complexity, autonomy, and late-discovery cost to set scope
"The model must be complete first."	Teams recreate large up-front specification	Use Candidate Specifications and iterate from uncertainty
"The simulation proves correctness."	Model limits and production difference are ignored	Document scope, fidelity, assumptions, exclusions, and residual risk
"The central team owns behavior."	Specialist queues weaken product and engineering accountability	Federate ownership; use the center for enablement and difficult shared problems
"Counterexamples are project failure."	People hide uncertainty and optimize demonstrations	Reward early contradiction and decisions improved by evidence
"More scenarios mean maturity."	Teams maximize counts instead of learning	Measure evidence sufficiency and outcome impact
"AI can approve what it generated."	Authority becomes circular and unaccountable	Retain human authority for intent, boundaries, material change, and risk
"The baseline freezes change."	Models become stale or teams bypass them	Treat every baseline as a versioned reference with explicit revision triggers

Table 14. Transformation safeguards preserve proportionality, learning, and accountability.

18.1 Cultural resistance

Resistance may reflect rational concerns: fear of bureaucracy, loss of autonomy, exposure of uncertainty, unfamiliar notation, or pressure on schedules. Leaders should respond with evidence and participation, not slogans. Show where a pilot changed a decision, remove avoidable friction, preserve team authority, and distinguish risk-proportionate practice from universal ceremony.

19. Leadership Playbook

19.1 Executives and sponsors

- 34. Connect SDSE to consequences the organization already recognizes.
- 35. Provide time, resources, authority, and cross-functional access for the pilot.
- 36. Use the evidence in a real decision and communicate what changed.
- 37. Protect teams that expose inconvenient assumptions early.
- 38. Demand proportionality and measurable value rather than artifact compliance.

19.2 Transformation leaders

- 39. Maintain the charter, roadmap, risks, dependencies, measures, and decision cadence.
- 40. Coordinate product, engineering, assurance, operations, training, and tooling.
- 41. Identify role-specific adoption barriers and remove them systematically.
- 42. Capture reusable learning and prevent the enablement function from becoming a gate.

19.3 Product and engineering leaders

- 43. Name the behavioral question and the decision authority before modeling begins.
- 44. Ensure the smallest sufficient model is created early enough to matter.
- 45. Link the baseline to architecture, implementation, tests, controls, and operations.
- 46. Maintain accountability for the behavior delivered and the risk accepted.

19.4 Team leaders and coaches

- 47. Create psychologically safe review of assumptions and counterexamples.
- 48. Facilitate participation from disciplines affected by the behavior.
- 49. Keep modeling practical, iterative, and decision-centered.
- 50. Turn each pilot into improved team skill and reusable organizational knowledge.

20. The First 90 Days

Period	Leadership and team focus	Expected evidence
Days 1-15	Confirm sponsor, define the problem, appoint transformation leader, identify decision candidates	One-page problem statement and sponsor mandate
Days 16-30	Assess readiness, select pilot, create charter, identify participants, establish baseline measures	Approved pilot charter and measurement plan
Days 31-45	Deliver targeted training, establish Workbench and repository, define ownership and review cadence	Pilot team capable of beginning real work
Days 46-60	Create Candidate Specification and initial BSN model; identify assumptions and priority scenarios	Reviewable candidate model and scenario portfolio
Days 61-75	Simulate, challenge, revise, and document evidence, limits, and alternatives	Decision-quality behavioral evidence
Days 76-90	Approve baseline, connect delivery work, compare measures, document lessons, decide next step	Stop, revise, repeat, integrate, or scale decision

Table 15. The first 90 days should produce a real decision and an evidence-based continuation choice.

90-day leadership question

What important decision is now better because the team could execute and challenge the proposed behavior before implementation?

21. The Integrated SDSE Transformation System

THE SDSE TRANSFORMATION SYSTEM

Lead the change through measurable pilots, lifecycle integration, distributed capability, and continuous learning.

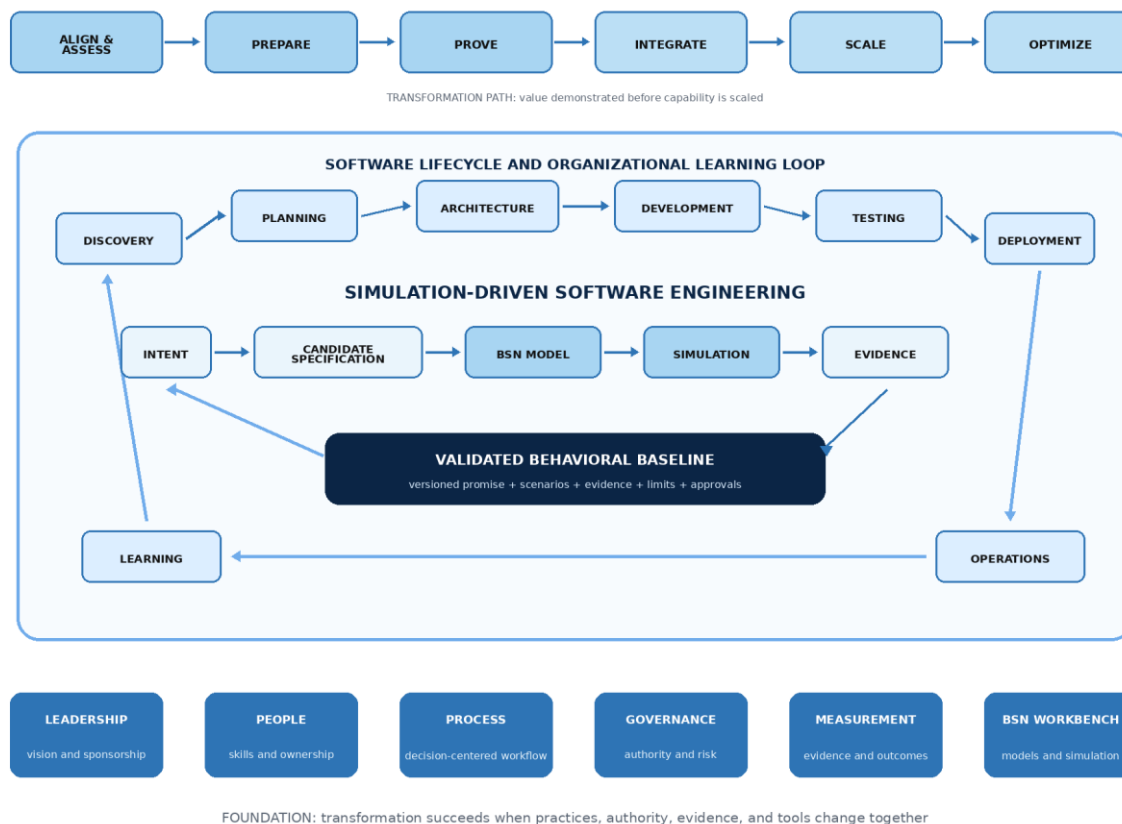


Figure 1. SDSE transformation joins staged organizational change, a behavioral evidence loop, lifecycle learning, and enabling foundations.

The figure shows one transformation system operating at three levels. The top path governs adoption: align and assess, prepare, prove, integrate, scale, and optimize. The middle system governs engineering: the SDSE evidence loop operates inside the full software lifecycle and returns production learning to intent. The foundation supplies the conditions that make both loops sustainable: leadership, people, process, governance, measurement, and the BSN Workbench.

The architecture deliberately places “prove” before “scale.” The organization first demonstrates that executable behavioral evidence changes a consequential decision. Integration then connects the validated baseline to existing delivery and operational systems. Scaling distributes capability through trained roles, reusable assets, and common evidence practices. Optimization improves model fidelity,

conformance, automation, portfolio learning, and AI assistance only where those investments create additional value.

Conclusion

Planning and leading SDSE transformation requires more than technical enthusiasm. It requires a clear problem, accountable sponsorship, a measurable value hypothesis, a bounded pilot, cross-functional participation, role-based capability, explicit decision rights, lifecycle integration, and the discipline to stop activity that does not improve worthwhile decisions.

The transformation should begin small but not trivial. One consequential item can demonstrate the complete logic: articulate intent, expose uncertainty, create a Candidate Specification, model behavior, simulate meaningful scenarios, review evidence, approve a baseline, connect that baseline to delivery, and compare the result with prior performance. When that chain produces credible value, the organization can scale through patterns, coaching, integrations, distributed ownership, and proportional governance.

The deepest change is cultural. SDSE makes uncertainty visible before it becomes expensive. Leaders must treat that visibility as progress. A counterexample found during simulation is not an obstacle to delivery; it is evidence that prevented the organization from committing more deeply to a flawed assumption. Teams that learn to value such evidence can move faster with greater confidence because their commitments rest on more than discussion and optimism.

Closing statement

The purpose of SDSE transformation is not to make organizations model-driven. It is to make consequential software behavior evidence-driven from intent through operations.

Appendix A. SDSE Transformation Readiness Assessment

Dimension	Readiness statement	Rating
Business problem	A recurring or material behavioral consequence is documented	1 2 3 4 5
Sponsor	An accountable leader will remove barriers and use the evidence	1 2 3 4 5
Decision	A specific choice can be changed by simulation evidence	1 2 3 4 5
Scope	A meaningful behavioral boundary can be completed in the pilot	1 2 3 4 5
Participants	Required product, engineering, assurance, and operations roles can engage	1 2 3 4 5
Culture	The team can expose and revise assumptions without blame	1 2 3 4 5
Information	Relevant policy, interfaces, incidents, and operating evidence are available	1 2 3 4 5
Tooling	Workbench, repository, access, and ownership can be established	1 2 3 4 5
Measurement	A baseline and comparison period can be defined	1 2 3 4 5

Table A1. Use ratings to identify pilot support needs; do not treat the total as an automatic go/no-go rule.

Appendix B. Pilot Selection Worksheet

Field	Working answer
Proposed use case	_____
Decision to improve	_____
Decision owner	_____
Late-discovery consequence	_____
Behavioral uncertainty	_____
Model boundary	_____
Priority scenarios	_____
Baseline measures	_____
Pilot duration	_____
Continuation decision date	_____

Table B1. Pilot selection worksheet.

Appendix C. Transformation Charter Template

Charter field	Required content
Purpose	Why this transformation matters now
Problem statement	Current condition, consequence, affected stakeholders, and evidence
Target capability	What people and teams will be able to do differently
Pilot scope	Included behavior, product, decision, period, and exclusions
Sponsor and leader	Accountable sponsor and day-to-day transformation leader
Participants	Required roles, teams, suppliers, and decision authorities
Value hypothesis	Expected leading, delivery, and business outcomes
Investment boundary	Time, training, tooling, facilitation, and integration limits
Risks and safeguards	Transformation, model, information, and operational risks
Decision cadence	Pilot reviews and stop/revise/repeat/integrate/scale date

Table C1. SDSE transformation charter template.

Appendix D. Roles and Responsibilities Matrix

Activity	Accountable	Responsible	Consulted
Approve transformation charter	Executive sponsor	Transformation leader	Product, engineering, finance
Select pilot	Sponsor / decision authority	Transformation leader	Product and engineering coalition
Define intent	Product authority	Product leader	Domain, engineering, assurance, operations
Own behavior model	Engineering or product leader	Behavior Model Owner	Cross-functional reviewers
Approve baseline	Named decision authority	Behavior Model Owner	Product, engineering, assurance, operations
Connect delivery artifacts	Engineering leader	Delivery team	Quality, security, platform, operations
Accept residual risk	Designated accountable authority	Decision owner	Assurance and affected stakeholders
Evaluate pilot	Executive sponsor	Transformation leader	Finance, product, engineering, team
Scale capability	Executive sponsor	Transformation leader / enablement center	Value streams and communities

Table D1. Adapt responsibilities to organizational policy while preserving explicit accountability.

Appendix E. Pilot Measurement Worksheet

Dimension	Definition	Working result
Behavioral clarity	Questions and contradictions resolved before implementation	Baseline: ____ Pilot: ____
Decision latency	Elapsed time from question to recorded decision	Baseline: ____ Pilot: ____
Planning	Estimate or forecast change after behavioral clarification	Baseline: ____ Pilot: ____
Flow	Blocked work attributable to behavioral uncertainty	Baseline: ____ Pilot: ____
Rework	Redesign, recoding, retesting, or integration correction	Baseline: ____ Pilot: ____
Quality	Behavior-related acceptance failures or escaped defects	Baseline: ____ Pilot: ____
Delivery	Behavior-related release delay, rollback, or recovery effort	Baseline: ____ Pilot: ____
Learning	Time from new evidence to approved behavioral response	Baseline: ____ Pilot: ____
Business outcome	Customer, financial, operational, policy, or trust result	Baseline: ____ Pilot: ____
Total investment	Training, facilitation, modeling, review, integration, and maintenance	Actual: _____

Table E1. Use comparable evidence and record confounding changes.

Appendix F. Maturity Assessment

Capability	Evidence statement	Assessment
Awareness	Leaders understand the evidence gap and can name relevant use cases	Not started / Emerging / Established
Experimentation	Pilots have used evidence in real decisions and measured outcomes	Not started / Emerging / Established
Repeatability	Multiple teams use defined roles, workflows, templates, and measures	Not started / Emerging / Established
Lifecycle integration	Baselines connect to delivery and operational evidence	Not started / Emerging / Established
Enterprise adaptation	Federated capability and reuse support cross-value-stream learning	Not started / Emerging / Established

Table F1. Assess only the maturity needed for the product and risk context.

Appendix G. Leadership Review Checklist

- 51.The problem and consequence remain clear and material.
- 52.The current modeling scope is tied to an accountable decision.
- 53.The pilot has the required stakeholder and cross-functional participation.

- 54. The team can explain assumptions, exclusions, fidelity, and model risk.
- 55. Counterexamples and unfavorable evidence are visible.
- 56. The decision authority has reviewed evidence and accepted residual risk.
- 57. The applicable baseline is linked to delivery and operational work.
- 58. Measures compare outcomes with an established baseline.
- 59. The enablement model is increasing team independence.
- 60. The next decision is explicitly stop, revise, repeat, integrate, scale, or optimize.

References

Sources were accessed on August 15, 2026. They support contextual claims about digital engineering transformation, workforce change, change leadership, software delivery, secure development, AI risk, software engineering knowledge, and Agile. The SDSE transformation framework is presented by the author.

- [1] Office of the Under Secretary of Defense for Research and Engineering. Digital Engineering Strategy, 2018. [Official strategy](#)
- [2] Office of the Under Secretary of Defense for Research and Engineering. Digital Engineering Workforce Plan, 2022. [Official workforce plan](#)
- [3] Kotter. The 8-Step Process for Leading Change. [Official methodology](#)
- [4] Prosci. Prosci Methodology Overview and ADKAR Model. [Official methodology overview](#)
- [5] Google Cloud. Announcing the 2024 DORA Report, 2024. [Official report announcement](#)
- [6] National Institute of Standards and Technology. SP 800-218, Secure Software Development Framework Version 1.1, 2022. [Official publication](#)
- [7] National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework. [Official NIST resource](#)
- [8] IEEE Computer Society. Guide to the Software Engineering Body of Knowledge, Version 4. [Official SWEBOK resource](#)
- [9] Beck, Kent, et al. Manifesto for Agile Software Development. [Official Agile Manifesto](#)
- [10] Hornbeek, Marc, with Debashis Bhattacharyya and Jitin Dhanani. Simulation-Driven Software Engineering: A Behavior-First Discipline for Modern Systems, 2026. [Purchase on Amazon](#)

About the Author

Marc Hornbeek, a.k.a. DevOps_the_Gray, is an award-winning engineer, author, educator, executive consultant, and keynote speaker with five decades of experience across systems engineering, packet networks, software development, behavioral simulation, testing, automation, DevOps, DevSecOps, SRE, platform engineering, and intelligent software delivery. His early engineering work included integrated tools for packet-network simulation, performance, conformance, observation, and test automation. He is the originator of Simulation-Driven Software Engineering and leads the development of Behavior Simulation Notation and the BSN Workbench. Marc received the IEEE Region 6 Outstanding Engineer Award in 2016 and has authored multiple books and professional certification courses. He founded PrimaModus® to help organizations model, simulate, validate, and engineer software that behaves as intended.

Contact: mhornbeek@engineeringdevops.com