THE EVIDENCE BASE

MEASURED, NOT IMAGINED

One organization. Two tracked owners. Full commit, issue, PR, review, and comment history — crawled weekly from GitHub, Dec 01 2025 through Aug 15 2026.

259 DAYS OBSERVED	93 REPOSITORIES	12,267 COMMITS	2,539 PULL-REQUEST CYCLES
2,597 CODE REVIEWS	1,825 HUMAN APPROVALS	11,143 REVIEW COMMENTS	1,786 ISSUES WORKED

RELEASE CYCLE = ONE PULL REQUEST CARRIED FROM INTAKE TO APPROVAL EVIDENCE. THE 1,000+ IN THE TITLE IS CONSERVATIVE BY 2.5×.

LEARNING OBJECTIVES

WHAT YOU'LL TAKE HOME

01 **A lifecycle map for embedding agents across the SDLC**
Nine stages, one agent role per stage — not a coding assistant bolted onto the IDE.

02 **The use cases that pay first**
Legacy analysis, test generation, documentation recovery, release evidence.

03 **The failure modes you will hit — and the control that answers each**
Hallucinated requirements, context drift, brittle tests, automation sprawl.

04 **A governance pattern that scales with speed**
Named human gates, traceability, evaluation harnesses, measurable release criteria.

05 **A phased roadmap from first agent to governed release factory**
Observe, assist, govern, scale, optimize — mapped onto our own adoption curve.

SECTION 01 · WHY MODERNIZATION STALLS

THE BOTTLENECK IS NOT CODE. IT IS CONFIDENCE.

Most legacy modernization programs are constrained by understanding, not effort.

SECTION 01

MODERNIZATION IS A DELIVERY PROBLEM

The challenge is not converting code from one platform to another. It is understanding, changing, testing, and releasing old systems fast enough — with proof.

CONSTRAINT 01

Decades of embedded business logic

Mission behavior lives in the code, not the specs. Nobody holds the full picture.

CONSTRAINT 02

Documentation is incomplete, outdated, or tribal

The people who knew are reassigned, retired, or were never asked to write it down.

CONSTRAINT 03

Manual modernization means long timelines and regression risk

Every refactor is a bet against behavior no one can fully enumerate.

CONSTRAINT 04

Mission systems cannot pause while you rewrite them

Continuity, security, and explainability are non-negotiable during the transition.

SECTION 01

FOUR FUNCTIONS. ONE TEST.

Modernization fails when a team cannot safely do one of these four things fast enough. Agentic AI earns a place only where it improves one of them.

UNDERSTAND

Recover business logic, dependencies, data flows, and system behavior from the code itself.

CHANGE

Propose refactors, generate migration paths, decompose work into reviewable increments.

TEST

Generate regression tests, edge cases, and validation artifacts before the change lands.

RELEASE

Assemble review evidence, documentation, compliance checks, and readiness gates.

If an agent does not improve **understand, change, test, or release** — it does not belong in the pipeline.

SECTION 01

THE REAL RISK

The risk is not that AI writes code. The risk is that AI writes code faster than your organization can validate it.



ILLUSTRATIVE PROPORTIONS — THE MEASURED VERSION OF THIS GAP FOLLOWS IN SECTION 02.

Do not deploy “AI developers.” Build an **AI-assisted release system**.

SECTION 02 · THE DATASET

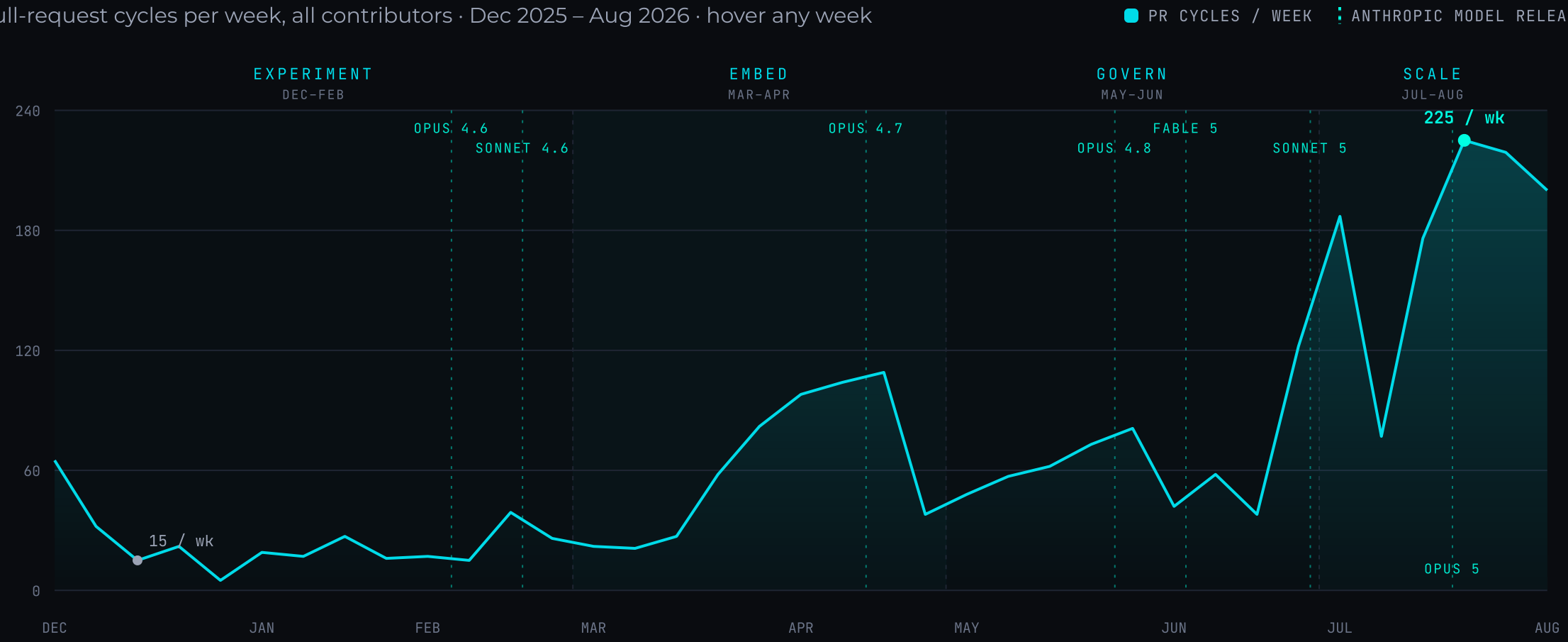
WHAT 1,000+ RELEASE CYCLES REVEALED

Repeated cycles expose the difference between impressive demos and durable delivery patterns.

SECTION 02

FROM AI EXPERIMENTS TO AI RELEASE OPERATIONS

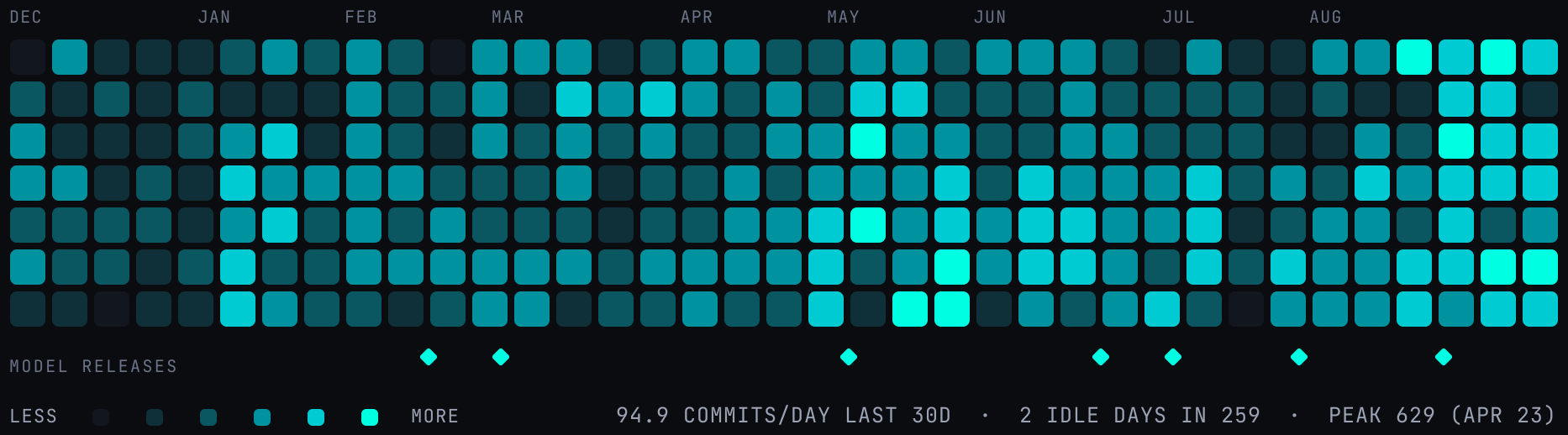
Pull-request cycles per week, all contributors · Dec 2025 – Aug 2026 · hover any week



SECTION 02

A FACTORY, NOT A SPRINT

Commits per day, every tracked repository · hover any cell · Sun–Sat columns, GitHub-style



SECTION 02

AGENTS PROPOSE. HUMANS APPROVE.

Contribution by lane across all 259 days — the split GitHub actually recorded

DRAI TEAM · AUTOMATION & AGENTS · UPSTREAM SYNC



SECTION 02

SAME TEAM. NEW OPERATING MODEL.

First 90 days vs most recent 90 days, all activity lanes



Review throughput grew **+240%** alongside PR throughput **+393%**. Speed did not outrun scrutiny — **scrutiny scaled with it**.

SECTION 02 · THE MONTHLY ARITHMETIC

FOUR MONTHS. THREE YEARS.

One 30-day month, measured at both ends of the curve — the same unit on both sides. Apples to apples.

x8.0	x9.2	x5.7	x7.1	x3.5
PR CYCLES	ISSUES	REVIEWS	APPROVALS	COMMITS
105 → **842** /mo	53 → **487** /mo	140 → **794** /mo	88 → **628** /mo	824 → **2,847** /mo

BASELINE = DEC 01–FEB 28 ÷ 3 — EXACTLY 3 × 30 DAYS · NOW = LAST 30 DAYS, JUL 17–AUG 15 — OPUS 5 (JUL 24) IN-WINDOW, FABLE 5 + SONNET 5 UPSTREAM

PROJECTION · SEP 01 – DEC 31 · CURRENT CADENCE HELD FLAT

4 MONTHS ≈ 3 YEARS

3,368 PR CYCLES = 32 BASELINE MONTHS (2.7 YRS) · 1,948 ISSUES = 37 (3.1 YRS)

AT THE OLD PACE, THIS WORK CLEARS IN 2029.

Assumes zero further improvement — no new model releases, no added scaling. The curve two slides back says that assumption is pessimistic.

SECTION 02

FIVE DURABLE PATTERNS

What survived 37 weeks of release cycles — and what quietly died.

---

01 **Small, testable units beat big-bang changes**

Agents are dramatically more reliable when work is decomposed before they touch it.

02 **One constrained role per agent beats one agent that does everything**

A test agent that only writes tests is auditable. A do-everything agent is not.

03 **The highest ROI lands before coding**

Analysis, documentation, test generation, dependency mapping — the unglamorous work.

04 **Human review works when structured around named gates**

More review is not the answer. More structured review is.

05 **Feedback loops beat one-off prompt experiments**

Release data flows back into prompts, agents, and tests — every cycle improves the next.

SECTION 03 · THE OPERATING MODEL

AGENTS BELONG IN THE WORKFLOW, NOT THE WILD

Map agents to the lifecycle — not onto the IDE.

SECTION 03

THE AGENTIC SDLC MAP

Nine stages. One constrained agent role per stage. A named output at every station.

S1	S2	S3	S4	S5	S6	S7	S8	S9
INTAKE	DISCOVERY	PLANNING	BUILD	TEST	SECURITY	DOCS	RELEASE	OPERATE
`Requirements agent`	`Legacy analysis agent`	`Decomposition agent`	`Refactoring agent`	`QA / test agent`	`Secure code agent`	`Documentation agent`	`Readiness agent`	`Feedback agent`
User stories, acceptance criteria	Code maps, dependency graphs, logic summaries	Work packages, modernization sequence	Proposed code changes	Regression tests, edge cases, coverage gaps	Vulnerability findings, policy checks	System, API, and release notes	Go / no-go evidence package	Defects, telemetry, improvement backlog

Each agent is **specialized, constrained, and measurable** — it holds one role, produces one class of output, and its work is evaluated at the next gate.

SECTION 03

IT RUNS WHERE YOUR TEAMS ALREADY WORK

Every agent output lands in a native GitHub object — reviewable, auditable, versioned. No parallel universe of AI artifacts.

AGENTS WRITE INTO

Issue	Branch	Commits	Checks	Draft PR	Review	Merge	Release
Backlog decomposition	Branch + commits	Generated tests	CI checks	Scan findings	Draft PR + evidence	Review comments	Release notes
EPIC → tasks, acceptance criteria	One work package	Small, traceable units	Tests · SAST · deps	Evidence package	Structured human gate	Protected branch	Notes + evidence archive

HUMANS HOLD

Requirement sign-off	Gate criteria	Risk acceptance	Review + approval	Merge authority	Release authority	

SCREENSHOT SET A · ISSUE → PR → CHECKS — SEE FACILITATOR GUIDE

SECTION 04 · WHAT WORKED BEST

THE BEST AGENTS MAKE THE INVISIBLE VISIBLE

The strongest use cases reduce ambiguity and increase verification.

SECTION 04

SIX PATTERNS THAT PAID FIRST

01 · LEGACY CODE EXPLANATION

Modules, dependencies, interfaces, and business rules — summarized on demand, cited to source.

02 · DOCUMENTATION RECOVERY

Living documentation rebuilt from code, tickets, tests, and deployment history.

03 · TEST GENERATION

Unit, integration, regression, and edge-case tests produced before refactoring begins.

04 · REFACTORING SUPPORT

Large changes broken into reviewable increments with dependency awareness.

05 · RELEASE EVIDENCE GENERATION

Traceability matrices, change summaries, and validation artifacts assembled per PR.

06 · TECHNICAL DEBT TRIAGE

Debt ranked by mission impact, change frequency, complexity, and risk — not by opinion.

Note what leads: **four of six are understanding and verification**, not code generation.

SECTION 04

THE DRAFT PR IS THE EVIDENCE PACKAGE

The pull request stops being a merge request. It becomes the release-confidence artifact.

DRAFT #1287 · Refactor auth session store — work package 3 of 5

Linked issue & acceptance criteria
Closes #1240 · 6 of 6 criteria mapped to diffs and tests

Impact summary
3 files · 2 functions · 1 schema touched · downstream consumers enumerated

Test results
412 passed · 0 failed · 14 new regression tests · mutation score reported

Security findings
SAST clean · 1 dependency bumped · 0 critical, 0 high open

Deployment & rollback
Staging deploy green · rollback path documented · feature-flagged

Reviewer guidance
Where to look first, known trade-offs, questions for the human gate

Checks · required to merge

unit + integration tests	PASS · 412
e2e user journeys	PASS · 37
SAST / code scanning	PASS
dependency + container scan	PASS
traceability check	6/6 mapped
evidence package complete	PASS
human review + approval	REQUIRED

● LIVE DEMO 02 · A REAL DRAFT PR

SECTION 05 · WHAT FAILED

AI DOES NOT REMOVE RISK. IT MOVES RISK UPSTREAM.

The failure modes are predictable enough to govern.

SECTION 05

PREDICTABLE FAILURES, PAIRED CONTROLS

FAILURE MODE WE HIT	THE CONTROL THAT ANSWERED IT
Hallucinated requirements	Source-grounded requirements traceability — every requirement cites its origin
Overconfident code changes	Mandatory test + review gates — no green checks, no merge, no exceptions
Context drift on long tasks	Bounded context windows and retrieval controls per agent role
Brittle generated tests	Mutation testing and coverage review — tests must fail when the code breaks
Security blind spots	SAST, dependency and container scans, policy-as-code in every pipeline
Inconsistent agent outputs	Prompt and agent-config version control plus evaluation harnesses
Automation sprawl	Role-based agent permissions and audit logs — agents get least privilege too

SECTION 05

TRUST IS EARNED AT EVERY GATE

Governance is embedded in the pipeline — not a paperwork step after the fact.

Gate 1

Requirements approved

Human validates intent + traceability before build

Gate 2

Architecture approved

Change fits the system — dependencies and contracts hold

Gate 3

Security cleared

Scans pass, policy-as-code satisfied, exceptions signed

Gate 4

Release authorized

Evidence package complete — a person owns the go

REQUIREMENT → CODE → TEST → EVIDENCE → RELEASE NOTE

One unbroken traceability chain per change — queryable in both directions.

PROMPT + AGENT CONFIG VERSIONED

AUTOMATED OUTPUT EVALUATION

ROLE-BASED AGENT ACCESS

AUDIT LOGS ON EVERY ACTION

QUALITY MONITORED PER RELEASE

SECTION 05

MEASURE SAFE CHANGE VELOCITY

Not lines of generated code. Release quality and modernization throughput.

SPEED

Cycle time · lead time · story throughput

QUALITY

Escaped defects · regression failures · rollback rate

MODERNIZATION

Debt retired · services refactored · cloud-readiness score

SECURITY

Vulns introduced · policy exceptions · remediation time

GOVERNANCE

Traceability completeness · review pass rate · evidence quality

DEVELOPER EXPERIENCE

Time-to-understand legacy code · documentation freshness

1.02 REVIEWS / PR 72% PRS WITH EXPLICIT APPROVAL 100% OF APPROVALS HUMAN 2 IDLE DAYS / 259

SECTION 06 · THE PLAYBOOK

CRAWL. WALK. RUN. THEN AUTOMATE THE SIDEWALK.

Start with bounded, repeatable workflows. Expand only on evidence.

SECTION 06

THE FIVE-PHASE ROADMAP

The same phases you watched on the adoption curve — dated from our own telemetry.

P1

OBSERVE

OUR DEC-FEB

- Code explanation on demand
- Documentation recovery
- Dependency mapping
- Test-gap analysis

P2

ASSIST

OUR MAR-APR

- Proposed refactors
- Generated tests
- Draft release artifacts
- Backlog decomposition

P3

GOVERN

OUR MAY-JUN

- Policy gates in CI
- Evaluation harnesses
- Prompt versioning
- Traceability controls

P4

SCALE

OUR JUL-AUG

- Reusable patterns across repos
- Parallel work packages
- Portfolio-level rollout
- 225 PR cycles / peak week

P5

OPTIMIZE

CONTINUOUS

- Release data tunes prompts
- Agents and tests improve
- Modernization resequenced
- Every cycle feeds the next

The order matters: **governance before scale**. Reversing P3 and P4 is how AI technical debt gets minted.

SECTION 06 · DEPARTMENT OF THE AIR FORCE

MISSION VELOCITY REQUIRES MISSION ASSURANCE

Assurance is the constraint that matters

Mission systems demand reliability, security, and explainability. Any acceleration that erodes them is negative progress.

Modernization cannot pause operations

The factory pattern delivers change in small, evidenced increments — continuity is a design property, not a hope.

Agents strengthen the evidence trail

Traceability matrices, test artifacts, and scan results are generated per change — the material assessors and authorizing officials actually need.

The outcome is trusted capability, faster

Not faster coding. Faster delivery of capability you can defend — for cloud migration, cyber resilience, and technical-debt reduction.

CLOSING

FIVE LESSONS FROM 1,000+ CYCLES

01 Agents work best when specialized, constrained, and measurable.

02 The biggest gains come from understanding, testing, and documentation — not just code generation.

03 Human review must become more structured, not merely more frequent.

04 Every AI-generated change needs traceability, tests, and release evidence.

05 The future SDLC is not autonomous development. It is governed human-agent collaboration.

▼ THIS IS THE SLIDE YOU MAY WANT TO PHOTOGRAPH ▼

THE MINDSET

THE FUTURE IS NOT AI REPLACING DEVELOPERS. IT IS DEVELOPERS COMMANDING DELIVERY SYSTEMS.

THE AGENTIC DELIVERY EXPORTABLE

Everything in this session ships as a deployable kit: agent role definitions, gate templates, policy-as-code, evaluation harnesses, PR evidence formats, and the telemetry panel you watched live — ready to stand up inside your environment. Ask for the exportable brief.

CONTACT

Corey Solivan

corey@product-ties.com · 703-955-0326

Data Room AI · TIE LLC

ATDC · C/O Data Room AI · 75 5th St NW, Suite 2310 · Atlanta, GA 30308

CONNECT

youtube.com/@coreysolivan

linkedin.com/in/corey-solivan

product-ties.com

draics.com

draics.com/DAFITC26 · this brief

aqtx1.substack.com · AQtx

What's possible today · what's coming tomorrow · what's stopping us now.

CONTRACT

OTA · SBIR · CSO

corey@draics.com

Rapid-acquisition pathways — vehicle mapping in the exportable brief.

QUESTIONS — 5 MIN