carahsoft

INDIRECT TEK

The Origins of Elixium

How two decades of building software the hard way taught us to treat AI agents as teammates

---

Thank you for your interest in exploring this content.

Carahsoft is the **Trusted Government IT Solutions Provider**® supporting a broad portfolio of industry-leading technologies through NASA SEWP V and a wide range of other contract vehicles.

As the **Master Government Aggregator**®, Carahsoft connects government agencies, industry partners, and technology providers to deliver innovative, mission-focused solutions.

In partnership with IndirectTek, we provide technology solutions that drive modernization, strengthen operations, and ensure compliance with evolving government standards.

To learn more about how Carahsoft can support your technology needs, please visit carahsoft.com

Explore More Resources:
carah.io/IndirectTekResources

Join Events & Webinars:
carah.io/IndirectTekEvents

Discover Technology Solutions:
Carah.io/IndirectTek

Learn About Procurement:
Carah.io/IndirectTekcontracts

Connect With Our Team:
IndirectTek@carahsoft.com
(571) 662-3222

The Origins of Elixium.ai

How two decades of building software the hard way taught us to treat AI agents as teammates

By Bradley Reid and Paul Beccio, Founders, IndirectTek

Elixium.ai is the AI-native agile platform built for the era when AI agents are teammates. The same guardrails that kept human teams shipping good software, now applied to the agents working alongside them. You might be surprised to learn Elixium did not start as a product idea. It started as a frustration we had both carried for years, from different seats, in very challenging Federal environments.

We met inside one of them. Brad was a Pivotal support engineer. Paul was leading a federal team at the Department of Homeland Security trying to do something genuinely difficult: not just adopt Cloud Foundry, but adopt the whole Pivotal Labs methodology. Extreme programming. Test driven development. Pair programming. Continuous delivery. The full discipline, inside a federal agency, with all the institutional resistance that implies. For two and a half years we were in the trenches of real delivery together. What came out of it was a shared understanding of what good software delivery actually looks like, and a shared frustration with how rarely organizations are willing or able to operate that way.

That shared understanding is the seed of Elixium. This is the story of how it grew.

What Cloud Foundry taught us about inheritance

There is a moment that shaped everything that came after, and it fit inside two words: **cf push**

Behind that one command sat a thousand solved problems. Build process, containerization, routing, secrets management, identity federation, environment promotion, health management, audit logging, compliance controls. All of it abstracted away. Solved once, inherited by every application that ran on top. The developer's job was to write code. The platform's job was everything else.

Most people who saw the demo thought they were watching a convenience feature. A few understood they were watching a paradigm shift. The deepest version of that shift was something the government technology world took years to name. Because Cloud Foundry absorbed the security controls into the platform itself, applications running on it inherited those controls automatically. The platform was accredited. The application inherited the accreditation. A two year Authorization to Operate could compress into weeks. Not because corners were cut, but because the platform had already done the work.

That idea, inheritance of hard-won discipline so that the people on top do not have to solve it again, never left us. Hold onto it. It is the spine of Elixium.

The pattern we kept seeing

After Pivotal, through the VMware acquisition and then the Broadcom acquisition, we both moved through other organizations. Different companies, different contexts, the same observation made over and over.

Organizations could see the feature. They could not see the value. They could adopt the tool. They could not adopt the philosophy. They could execute the process. They could not internalize the why behind it.

The weekend proof of concept always looked amazing. The production reality was always harder than anyone admitted. And the institutional knowledge that made the difference was always sitting in the heads of a small number of people who had been doing this long enough to have already made every mistake. Those people were consistently undervalued, misunderstood, or scattered by the next round of acquisitions.

We had seen this movie before. So when AI assisted development arrived, we recognized the opening scene immediately.

We have seen this movie before

AI is doing to the path to production what Kubernetes did to platform engineering.

The parallel is not superficial. It is structural. A powerful new capability arrives and compresses what used to be hard into something that looks easy on a Friday afternoon. Organizations mistake the speed of the demo for the readiness of the production reality.

The governance infrastructure built for the old pace becomes a bottleneck at the new pace. And the people who see it early are the ones who lived through the previous transition and remember what the Monday morning after the weekend proof of concept actually looked like.

A developer can now generate five hundred lines of code in minutes. That is real, and it is genuinely useful. But the old pipeline cannot govern at AI speed, and nobody had built the new one. That gap is where Elixium lives.

The Monday morning question

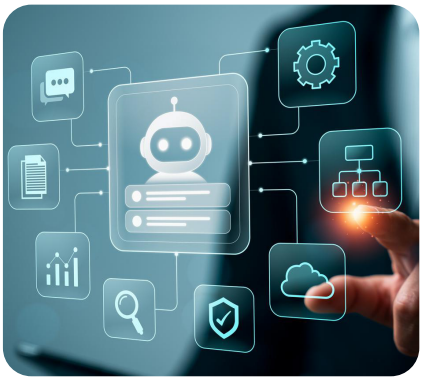
With Kubernetes, the question was:

- Where is the service mesh, the secrets management, the role based access control?

With AI, the question is sharper and more uncomfortable:

- Where did this code come from?
- Is it compliant?
- Who owns it?
- Does it reflect how we actually work, or did an agent just confidently invent its own approach?

Why the Pivotal practices matter more now, not less



Here is the conclusion that surprised even us. The arrival of capable AI agents does not make the old disciplines obsolete. It makes them load-bearing.

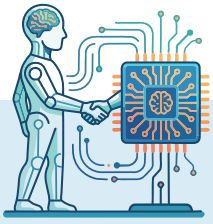
Think about what a Balanced Team was always for. The Pivotal model put a product manager, a designer, and an engineer together so that desirability, viability, and feasibility informed each other continuously, rather than getting bolted on at the end. The model worked because every member shared a vision, sat in the same space, communicated constantly, and understood the why behind the work. Build, measure, learn. Small vertical slices. Stories as placeholders for a conversation. Sustainable pace so the team could go fast forever. These were never bureaucratic ceremonies. They were the mechanisms that kept a team aligned, honest, and fast under uncertainty.

Now put an AI agent on that team.

An agent can write code at a speed no human pair can match. But an agent does not arrive knowing your glossary, your prior decisions, your module boundaries, your interface contracts, or your definition of done. Left ungrounded, it does exactly what the ungrounded weekend proof of concept always did: it produces something that looks finished and is not safe to ship. It optimizes for output when the team needs validated learning. It says yes to everything when good product work has always been about knowing when to say no.

The disciplines that kept human teams aligned are precisely the disciplines an agent needs in order to be a teammate rather than a liability. The practices did not become less important. The stakes of skipping them went up.

Agents are part of the Balanced Team, not an afterthought



This is the conviction Elixium is built on. **AI agents are teammates** who need the same governance guardrails human teams use.

That is not a metaphor we reach for in marketing. It is an engineering position. If an agent is going to participate in delivery, then it has to participate in the team's shared understanding the same way a new human hire would. It needs to know how we work, and it needs to believe in it, in the only sense that matters for a system: its behavior has to be shaped by that understanding at the moment it acts.

So we did not design Elixium as a place where agents run loose alongside a human process happening somewhere else. We designed agents into the Balanced Team from the first principle. The same way a Pivotal team blends a product manager, a designer, and an engineer so their perspectives inform each other, Elixium blends human judgment and agent execution inside one grounded, governed workspace. The agent is not a second thought bolted onto an existing tool. It is a team member who shows up already aware of the vision, the decisions, and the boundaries, because the platform makes that awareness unavoidable.

How we practice it: governance and guardrails that teach the why

Remember the inheritance model. This is where it returns.

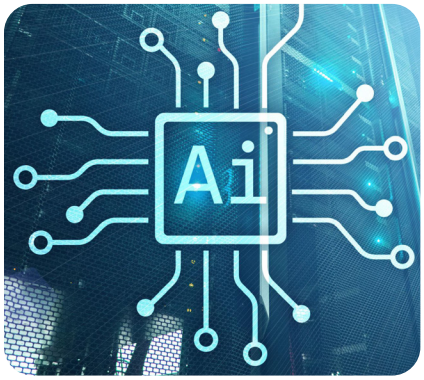
Elixium carries a persistent grounding layer that governs how agents behave. A glossary so the team and its agents share one vocabulary. A map of the modules and how they fit.

A record of the decisions the team has already made and why. The interface contracts that hold the system together. An agent operating in Elixium does not have to rediscover any of that, and it is not free to ignore it. It inherits the team's accumulated understanding of the way a Cloud Foundry application inherited the platform's controls. Solved once, available to every teammate that follows, human or agent.

Two principles shape how that grounding is enforced, and both come straight out of the scar tissue.

1. The first is that guardrails have to teach the reason, not just the rule. The most durable thing we ever did at Pivotal was transfer the why, not the procedure. A teammate who understands why a practice exists can generalize it to a situation no checklist anticipated. So Elixium does not hand its agents brittle scripts that say call this, then that, then the other thing. It frames the state the agent must be in. Be grounded before you write. An agent that understands that intent applies it everywhere, the same way a developer who internalized test driven development did not need to be told to write the test on every single story.

2. The second is that for the things that genuinely cannot be left to judgment, surfacing a state is not enough. A warning that an agent is free to override is documentation, not policy. Real governance means the safe path is the path of least resistance, enforced by the system itself. This matters most for compliance. In a federal environment, the cost of getting it wrong is not a sprint retrospective. So compliance controls in Elixium refuse by default rather than warn and allow. The discipline is built into the platform, not left to the good intentions of whatever is acting at two in the morning.



That is the whole point. We are not asking teams to trust that an agent will behave. We are building the environment that makes behaving the natural thing to do, and we are doing it by encoding the practices we spent twenty years learning to respect.

Elixium is not a tracker

We want to be direct about this, because it is the easiest thing to assume and the most important thing to get right.

Elixium is not a backlog with an AI bolted on. The old tools were built to organize human work into stories and move them across a board. That was useful, and a generation of teams, ours included, got real value from it. But a board does not know your decisions. It does not hold your vocabulary. It cannot tell an agent why a boundary exists, and it certainly cannot refuse to let one cross it.

Elixium is the grounded, governed environment where humans and AI agents do the work together, where the team's shared understanding is a living substrate that shapes how every teammate acts, and where the disciplines that made great software teams great are encoded so that agents inherit them rather than ignore them. It is the answer to the Monday morning question, built before the traffic jam forms.

The through line

From cf push to Elixium, the idea has not changed. Solve the hard things once. Let the people, and now the agents, on top inherit that work so they can move fast without moving blind.
Transfer the why, not just the what. Build the governance into the platform so that doing it right is the easy path.

We were there when the last paradigm shifted. We are here now because we see the next one coming, and this time we are not going to watch the institutional knowledge scatter. We are building it into the platform.

That is the origin of Elixium. The rest is what we build with you.

IndirectTek.com

Built on the Robinhood Principle: enable the people, give them the real answer, and trust that doing right by the client compounds in a way that doing right by the quarter never does.