Xiaotime Labs

A XIAOTIME WHITEPAPER · THE ARCHITECTURE

The Torus.

The shape of every running system.

Every system has a shape. A running system has one shape in particular.

PAPER	FROM	AUDIENCE
A Xiaotime Whitepaper	**Xiaotime Labs**	**CISO · CIO · Security leaders · Board**

A document is flat. It has a beginning and an end. You open it, you read it, you file it. Its shape is a line, or at best a plane. It describes a moment. It does not run.

A pipeline is different. A pipeline loops. What goes in comes back around, changed, and goes in again. There is no first step that stays first and no last step that stays last, only the system moving through itself.

The shape of a pipeline is a torus.

This paper is about that shape: what it is, why it keeps turning up wherever people describe things that sustain themselves, and why it is the shape of every running system we build.

The torus is made of three loops turning together: agents that act, people who judge, and an environment that answers back. Each change the agents make is shaped by human judgment and tested against what the environment reports, and what comes back becomes the input to the next turn. Xiaotime runs its security program, its software delivery, its design system and its publishing on those loops. Run them long enough and something new appears: a system that senses its own state, keeps a record of itself, and corrects itself.

A document is flat

Most of what organizations call governance has the shape of a document. A policy is approved on a date. A risk register is reviewed in a quarter. An assessment is performed in a window, signed, and filed. Each one is accurate on the day it is finished, and each one starts ageing the next morning.

That is not a failing of the people who write them. It is the shape. A flat thing has edges: a top where the work begins and a bottom where it is filed. It cannot loop, so it has to be rebuilt by hand, and the cost is the annual cycle: the scramble before the audit, the quiet after it, and the drift in between.

A control drifts in March. The document finds out in November.

The shape of something running

A torus is a ring: the surface you get by sweeping a circle around an axis beside it (Weisstein, MathWorld). It can also be built from a flat square. Join the top edge to the bottom to make a tube, then join the tube's two ends. Nothing is cut or twisted, and the square is now a torus (Weisstein, MathWorld; nLab).

Every straight lane on the square becomes a circle on the torus. A pipeline drawn as a straight line of stages becomes a loop the moment its ends are joined.

The edges are gone. Walk off the right side and you come back on the left; walk off the top and you return at the bottom. The surface is finite, and there is nowhere to fall off.

The flat version of the torus is also a group: any point can be carried to any other by sliding the whole surface, which lands exactly on itself (nLab). No point is privileged. Every point is as much the middle as any other.

Those are three properties a document cannot have and a running system must have:

- *No edge.* The work does not end at a boundary and wait to be restarted. It comes back around.
- *No privileged point.* No single moment, such as the audit or the annual review, carries the whole system. Every point on the loop is a point where the system is checked.
- *A hole that is part of the shape.* Fill the hole and you have a disc. The opening through the middle is what lets the surface close on itself.

Where the torus turns up

People who study things that sustain themselves keep reaching for this shape. The claims below are not equally settled, so each is stated at its real strength.

The shape of space

One model cosmologists test is a flat universe shaped like a three-dimensional torus: finite, with no edge. Such a universe would repeat patterns in the oldest light we can see. The Planck mission looked for them and found no such topology smaller than the observable universe (Planck Collaboration, 2016), and a later study argues many possibilities remain open (Akrami et al., 2024). The torus is not the established shape of space. It is a shape physicists take seriously enough to test, because it describes something bounded and continuous at once.

Holding a star in a ring

A tokamak, the leading design for magnetic fusion, holds plasma at roughly 150 to 300 million degrees Celsius in a ring-shaped chamber; its name is a Russian acronym for "toroidal chamber with magnetic coils" (ITER). Coils steer the plasma around the ring so it never touches the walls. A tube would have ends to escape from. A ring has none.

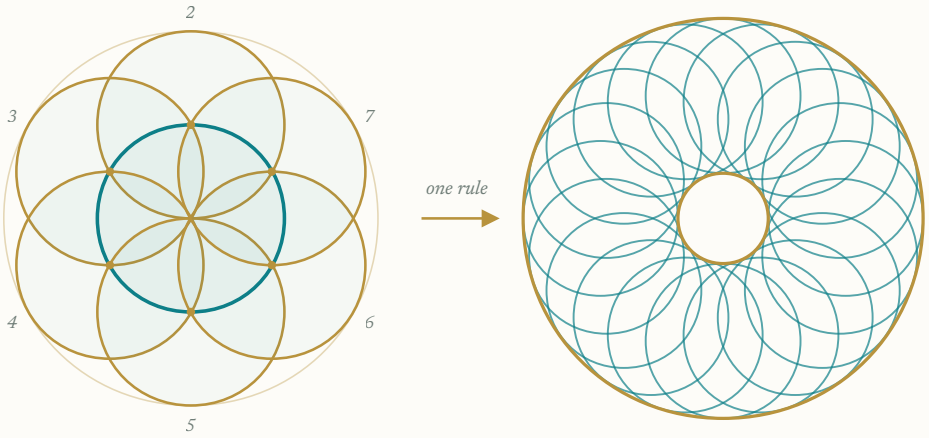
A map inside the brain

The strongest evidence is recent. In 2022, researchers recording hundreds of grid cells at once found that their joint activity lies on a torus. Where the activity sits on it tracks where the animal is, and each cell kept its place on it across environments and through sleep (Gardner et al., 2022). Part of the brain's sense of where it is runs on a torus.

The geometry of generation

Draw a circle. Set the compass on its edge and draw another of the same size. Step around, drawing a new circle where the last two cross. Six circles fit exactly around the first; the last one closes the ring, making the figure now called the Seed of Life. Slide a circle's centre around a ring and the overlapping circles trace a band around an open middle: a torus seen from above, the shape of a ring of spheres fused into a tube.

Both figures are one move repeated, with no privileged circle once complete, each growing by applying its rule to its own output. That is the property a running system needs: one rule, applied continuously, producing the whole.



The Seed of Life A torus, from above

The Seed of Life, built with a compass in seven steps, beside a torus seen from above. Both are one circle, repeated by a single rule around a centre that stays open.

Your security program has this shape, or it should

An information security management system was never meant to be a document. The standard that defines it uses four verbs for what an organization does with it: *establish, implement, maintain and continually improve* (ISO/IEC 27001, clause 4.4). All four describe motion. NIST's guidance on continuous monitoring, published in 2011, treats the program the same way, as an ongoing view of assets, threats and whether controls are working (NIST SP 800-137).

Somewhere along the way the loop was handed over as a template. The document-based ISMS has a centre: the annual review, the audit event, the moment when everything converges, gets checked, gets filed and goes quiet for another year. That centre is also a stop.

A running system has no centre. On a torus, every point is as much the middle as any other. The estate feeds the pipeline. The pipeline produces evidence. The evidence informs the controls. The controls shape the estate. Nothing waits for a date to be checked, because every pass around the loop checks it.

This is what Xiaotime means by *ISMS as Code*: the management system declared as code, enforced continuously and evidenced continuously. Most tools help you describe your ISMS. We are your ISMS, and the torus is its shape.

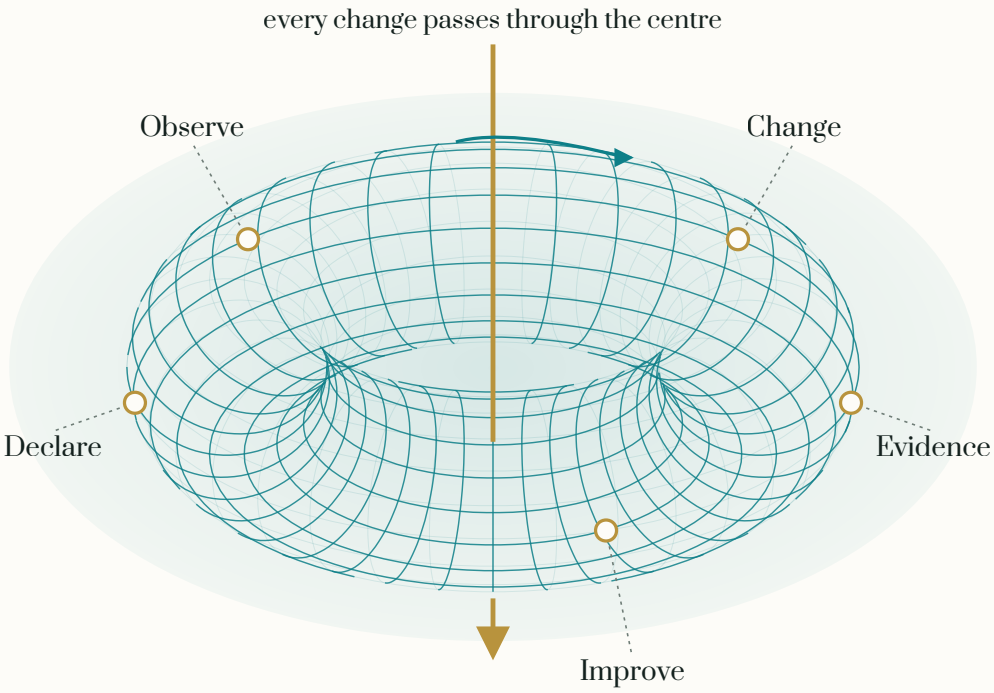
The hole is the pipeline

Look at a torus and the eye goes to the hole. It reads as something missing.

It is not missing. The hole is architecture. It is the opening the surface turns through on every pass, the place where outside becomes inside. Remove it and the surface cannot close on itself: you have a disc. A disc is a document. A disc does not run.

In a running program, the hole is the pipeline. Every change passes through it: code, infrastructure, configuration, a policy edit, a new agent, a design token, a paragraph of this paper. On its way through, the change is checked against what the organization has declared, approved by the people who own it, and recorded as evidence. Then it comes back out as part of the estate, and the next change starts its own pass.

GRCDevSecOps is the discipline that puts governance, risk and compliance inside that opening, in the same pipeline as development, security and operations. Not bolted on after the change. Running through the same gate as the change. ISMS as Code is the category; GRCDevSecOps is how it runs.



no first stage, no last stage

The torus pipeline. The estate is declared and observed on the outer surface, every change passes through the centre, and evidence returns to the surface as the loop closes. There is no first or last stage.

One turn of the torus

Follow one change all the way around.

Declare. The baseline, written as code, says a storage bucket holding customer exports is private.

Observe. On a Tuesday afternoon that bucket becomes readable by anyone. Nobody filed a ticket. The next capture of the estate says it is public. The difference is drift, and it is found on the next capture of the estate, not at the next review.

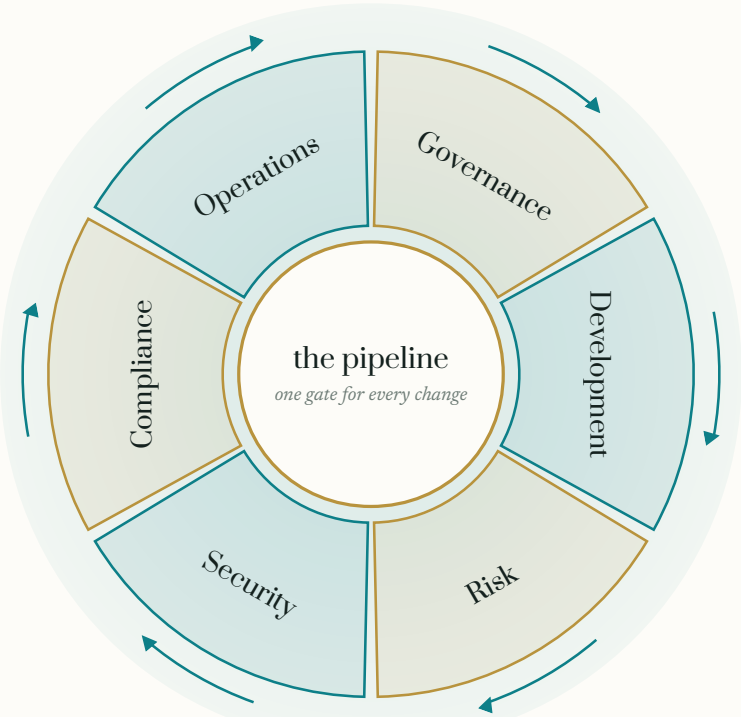
Change. An agent reads the declaration and the observed state side by side, drafts a pull request that restores the private setting, and attaches the controls the drift touches in every framework the organization answers to. The checks run on the proposal before any person sees it. Then the owner of the bucket reviews a small, specific change with its reason and its blast radius attached, and approves it. The approval is the review itself, recorded where the change lives.

Evidence. The merge applies the fix. The drift, the proposal, the checks, the approver and the times are kept as one signed record. When an auditor later asks whether that control was in place and since when, the answer already exists, including the hour it was not.

Improve. Fixing the bucket is half the turn. The other half runs in reverse: from the result back to its most likely cause. The record shows the setting was changed by hand, outside the pipeline, by a session holding a broad administrative role. That inference (reasoning backward from an effect to its best explanation, which security teams know as root-cause analysis) produces the second change: the declaration itself is tightened so that role can no longer change storage access by hand, and a change that tries is stopped or raised for an exception. That improvement goes around the loop like any other change.

The next turn starts from a better declaration. This class of drift cannot recur silently, because the system learned from the last time it did.

All three loops turned in that one pass. The environment answered (the bucket changed). An agent acted, fast and at machine speed. A person judged where judgment mattered. And the turn ran in both directions: forward from cause to fix, then back from result to cause, so that what the organization declares gets better each time around. Forward is close to intuition; backward is inference; the adjustment is learning.



gold: governance, risk, compliance · teal: development, security, operations

GRCDevSecOps mapped onto the torus. Governance, risk and compliance run around the same ring as development, security and operations, and all six pass through the one pipeline at the centre.

Everything as Code is one family

Infrastructure as Code won its argument years ago: servers written as code can be read before they exist, compared with yesterday's version and traced to whoever proposed the change. Everything as Code carries that move to every system an organization operates. Each discipline has the same shape, because each is the same loop applied to a different kind of thing.

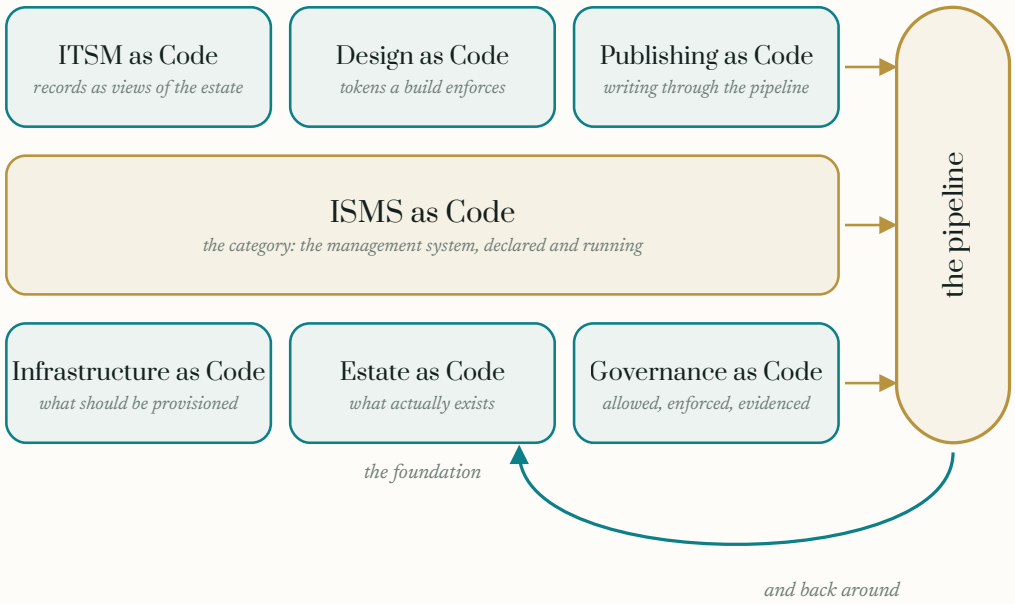
ISMS as Code is the management system itself: scope, risk method, policies, controls, monitoring, internal audit and review, declared and running. It is the category the rest of the family serves.

ITSM as Code treats service records as views of the estate, not a separate system to keep in sync. The configuration database is a read of what is declared and observed. A change is a pull request: the review is the approval and the merge is the record. An incident is a scored finding raised from drift. The service desk still holds its records; the loop keeps them in step.

Design as Code keeps a product's visual system in the loop: colours, type and spacing are tokens defined once, and a screen that uses anything else fails the build.

Publishing as Code puts writing through the pipeline: source in a repository, voice and brand rules as checks, review as approval, and an engine that renders the result. This paper and its diagrams were made that way.

Underneath sit Infrastructure as Code (what should be provisioned), Estate as Code (what actually exists) and Governance as Code (what is allowed, enforced and evidenced).



The Everything as Code stack. Infrastructure, estate and governance as code form the foundation; ISMS, ITSM, design and publishing as code run on it; every layer passes through the same pipeline.

Each discipline is a layer of the torus

The family maps onto the loop directly. Each discipline is strongest at one point on the ring, and every one of them passes through the same centre.

Point on the loop	What happens there	The discipline that owns it
Declare	What should exist is written down as code	Infrastructure as Code, Design as Code, Publishing as Code
Observe	What actually exists is captured and compared	Estate as Code
Change	Every change passes the gate at the centre	Governance as Code, ITSM as Code (change)
Evidence	The record is produced by doing the work	Governance as Code, ITSM as Code (record)
Improve	What was learned feeds the next declaration	ISMS as Code, which holds the whole loop

ISMS as Code appears last because it is not one layer: it is the management system that owns the whole loop. The other disciplines are the surfaces it runs on.

Xiaotime builds Xiaotime on it

A claim about running systems should be shown running, and this is not a demonstration built for a reader. It is how we work. Our own security program runs on the platform: our controls are declared as code, our estate is captured and compared continuously, and our audit evidence is produced by the work. The pipeline that builds the platform governs its own code, with the same gates for every change whether a person or an agent wrote it, and no carve-outs for the team that built the gates.

The same loop holds the rest. A screen that reaches for a colour outside our design system does not merge. This paper was written as source, checked against our voice and brand rules on every change, reviewed, and rendered by the engine that renders all our papers.

That is recursive integrity. The system governs the team that builds the system.

What falls out of the geometry

Once the program has the shape of a torus, several things stop being projects and become properties.

- *Audit readiness is not a sprint.* It is a property of a system that produces its evidence on every pass.
- *The next framework is cheap.* A new framework is a new reading of the same loop, not a new loop. Manual governance has no leverage; the Nth framework costs what the first one did. A torus has leverage.
- *Existing tools compound.* The scanners, monitors and detectors an organization already runs become the senses of the loop. They are wired in, not replaced.
- *Change is governed where it happens.* Not reported after the fact.
- *Drift has nowhere to hide.* A difference between what was declared and what exists is found on the next pass, not at the next review.

Three loops, read two ways

The turn above was driven by three loops, each with its own clock. *Agents act*, at machine speed: they read the estate, draft the fix and run the checks. *People judge*: they decide what the system is for, set its limits and approve what reaches production. *The environment answers*: accounts, devices, identities, code, users and attackers keep changing, and every change is a signal the next turn reads.

None of the three can run the system alone. Agents without signals have nothing to act on, and without people they have no limits. People alone cannot keep pace. The environment alone just drifts. Together they form one loop that keeps re-entering itself. There are two ways to read what that loop is.

One reading: an operating model. The loop is how a security program stays whole while the organization under it changes. Work that once waited for a review happens on every pass: drift is found when it occurs, fixes are drafted before anyone files a ticket, and people spend their judgment on the decisions that need it. Read this way, the torus is a description of good operations, and its value is measured in what it finds, what it fixes and what it proves.

Another reading: a system that knows its own state. In 2023, researchers in neuroscience, philosophy and machine learning turned the leading scientific theories of consciousness into *indicator properties*: features of how a system is built that can be checked against a real architecture. They found that no current AI system qualifies, and that nothing obvious in the engineering stands in the way (Butlin et al., 2023). Several of their indicators describe a well-built running system: processing that loops back on itself, a shared workspace that every part reads from and writes to, monitoring of its own states, and action steered by feedback. Read this way, the torus is the structure the science says to look for, and a loop built well enough moves closer to that edge than a document ever can.

Neither reading claims the system is conscious, or a general intelligence; nobody yet knows how either arises. One way to hold both: intelligence is throughput, how much a system can do; awareness is the loop, a system modelling itself as it models the world. The first reading is about the throughput. The second is about the loop.

Objections

""This is vendor marketing dressed as philosophy.""

Test it instead of taking it on trust. The torus makes claims that are either true of your program or not, and each can be checked in an afternoon.

- *Pick one control and ask when it was last checked, and how you know.* If the answer is a date in last year's assessment, the program has a centre and a stop. If it is the last pass of the loop, with its record, it does not.
- *Change something outside the process and time how long it takes to be noticed.* Make a harmless configuration change in a test account by hand. A running program finds it on its next pass. A document finds it at the next review.
- *Ask for the record of a recent change.* Who approved it, what it touched, which requirements it affects. If that record was produced by doing the change, the loop is real. If someone has to assemble it, it is not.

The shape is a way of seeing the structure quickly. The tests are what make the argument hold or fail, and they work the same whether or not you ever buy anything from us.

The convergence

Cosmologists test the torus as a space that is bounded and continuous. Fusion engineers use it to hold what must never touch its walls. Neuroscientists found it in the brain's map of where it is. Software engineers reach for it when they want a pipeline with no privileged centre and no stopping point. These are different observations that solve one problem: how to keep something whole while it keeps moving.

A running security program has the same problem. It has to stay whole while the organization under it changes every day, and it has to prove that it did. The shape that solves that problem is a torus, turned by three loops: agents acting, people judging, the environment answering. A system built that way senses itself, remembers itself and corrects itself.

Xiaotime built it. Our own security program runs on it, and so do our service records, our design system and this paper. The loop is running.

Read this as an argument. If it holds up against your environment, we should talk.

Related reading

- **ISMS as Code.** The security program, running as code. Not described. Operated.
- **Everything as Code.** The velocity case for moving governance into the code layer.
- **Estate as Code.** Capturing the whole estate, including the AI, as code you can govern.
- **GRCDevSecOps.** Why governance belongs in the pipeline, not the report.

Sources. Akrami, Y. et al. (COMPACT Collaboration), "Promise of Future Searches for Cosmic Topology," *Physical Review Letters* 132, 171501 (2024); arXiv:2210.11426. Butlin, P., Long, R., Elmoznino, E., Bengio, Y. et al., "Consciousness in Artificial Intelligence: Insights from the Science of Consciousness," arXiv:2308.08708 (2023). Gardner, R. J. et al., "Toroidal topology of population activity in grid cells," *Nature* 602, 123-128 (2022), doi:10.1038/s41586-021-04268-7. ISO/IEC 27001:2022, clause 4.4. ITER Organization, "What is a tokamak?", iter.org. NIST Special Publication 800-137, *Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations*, September 2011. nLab, "torus," ncatlab.org. Planck Collaboration, "Planck 2015 results. XVIII. Background geometry and topology," *Astronomy & Astrophysics* 594, A18 (2016); arXiv:1502.01593. Weisstein, E. W., "Torus," *MathWorld*, Wolfram Research.