

Fachhochschule Graubünden
University of Applied Sciences

AISE502: AI in Software Engineering II

Lecture 11: The Fit III – The Measurement Contract, Conway's Law, and the Limits of the Theory

Script: Part IV, Sections 37–39

Agenda

1. The **measurement contract** – step (vii)
2. Fitness functions: taxonomy and three instrument families
3. **DORA** and the coupling finding
4. The four-layer cascade; the C10 reference contract
5. Cost of change: Boehm vs. Menzies
6. AI Lens, both axes; the project's contract
7. **Conway's law** and Team Topologies
8. Limits of the theory; **Maxim 9**
9. **This week's exercise**: resilience complete

Recap: where we are

Part IV so far:

- **L7**: three cases, three stages, the procedure, the 7 × 10 matrix, the contract introduced
- **L8–9**: Part III, classes C1–C9
- **L10**: hybrids and evolution paths (Segment, Prime Video, Shopify), Maxim 5; the eight-step procedure with ADR-007, Maxim 6 (a response measure turns an opinion into a testable hypothesis); the ten matrix rows cell by cell – recurring resolution: *consistent core, asynchronous edges*
- Deck 1 (A5) and deck 2 (D9, D11) already named the four DORA metrics, the scaling finding, the *within*/*of* cost asymmetry and Conway's law – today the instrument taxonomy, the numbers and the evidence behind them
- The measurement contract is the **fifth framework element**; ADR-011 (lecture 3) already ended with an embryonic three-line contract – its lines are rows 2, 7 and 8 of today's reference contract; the static rule tightened from "no provider-SDK import" to "gateway only via the declared port"
- Deck 6: every cell is a **default hypothesis**, replaced by measurement once the system exists – today: how that measurement is organised (steps (vii)–(viii))
- Deck 6: *Conway is a decision filter, not a footnote*, and the evidence base has honest gaps – today §38 makes the organisation the third party to the fit, §39 turns the gaps into six limits the theory states about itself
- **Today closes Part IV; next week opens Part V**

How does a decision made this year stay honest in year five?

How does a decision made this year stay honest in year five?

Example: Prime Video – the trigger was a measurement

- What actually triggered the Prime Video re-architecture was **not an architecture review but a telemetry signal**: infrastructure cost per stream, measured continuously, crossed what the team was willing to pay
- That measurement, not an opinion, first *forced* and then *vindicated* the redesign
- The cost dashboard was a **fitness function in everything but name**: an objective, continuously evaluated check on an architectural characteristic whose breach converted a running structure from “accepted” into “falsified”

The empirical anchor for building such checks systematically is DORA: coupling – this theory’s leading dimension – is a **measurable** property – the finding itself comes with the delivery layer (37.2), after the three instrument families.

Step (vii): the fit becomes a measurement contract

- Step (vii) of the procedure generalises the Prime Video observation into a concept – and it is where this course differs from a classical architecture lecture
- The chosen fit is codified as a **measurement contract**: the set of *executable invariants* under which the architecture is allowed to keep evolving

Key Concept

"The architecture may change freely as long as the contract stays green."

What today adds to the week-7 introduction:

- the instrument taxonomy (37.1)
- the delivery layer – DORA (37.2)
- the four-layer cascade and the C10 reference contract (37.3)
- the economics – cost of change (37.4)

Architectural fitness function – the definition

Definition: Architectural fitness function

An architectural fitness function is “any mechanism that provides an objective integrity assessment of some architectural characteristic”. Fitness functions turn quality attributes into **executable, objective checks** and thereby move architecture governance from review meetings into the CI/CD pipeline.

Classified along two primary dimensions – **scope** and **cadence**:

Scope	*atomic*: one characteristic in isolation, e.g. a dependency rule as a unit test	*holistic*: combined characteristics in interplay, e.g. security and data freshness under load	
Cadence	*triggered*: event-based, on every build or deployment	*continual*: running permanently in operation, e.g. chaos experiments	*temporal*: time-scheduled, e.g. dependency-freshness time bombs

Instrument family 1: dependency checks as CI gates

- **ArchUnit** (Java); analogues: NetArchTest (.NET), dependency-cruiser (JavaScript), import-linter (Python)
- Rules formulated as **unit tests that fail the build on violation**:

  “the domain layer imports no framework”
  “no cycles between modules”
  “repositories are called only by services”

- **Spring Modulith verification** does the same for module boundaries *declared* in a modular monolith
- Classification: **atomic, triggered**
- For the matrix: this is what makes the MM ratings of the capability table **enforceable rather than aspirational**
- (recap, deck 4) without automated boundary verification, *boundary erosion* is the documented failure mode of the pattern – “the first fitness function most teams ever write”

Instrument family 2: performance and cost budgets as pipeline gates

Classical budgets

- latency thresholds, bundle sizes, or Lighthouse scores
- declared in a **budget file**
- gate the pipeline (Lighthouse CI)

$\Longrightarrow$ same mechanism

Axis-B budgets

- **token-cost budgets** per AI use case
- **p95 latency budgets** per AI use case
- the same mechanism with *new units*

The transfer to Axis B is direct. In the C10 reference contract the units read:

- advisory answer p95 $< 20\,\text{s}$ end-to-end (triggered + continual)
- token cost $\leq$ budget, e.g. CHF 0.40/request at p95 (continual)

Instrument family 3: chaos experiments (continual, holistic)

- Netflix's **Chaos Monkey** terminates production instances to test resilience assumptions *permanently*
- Formalised as the *principles of chaos engineering*; cited by Ford et al. as the paradigm of a **continual, holistic** fitness function
- (recap, deck 2 D5) for the matrix: chaos experiments are the instrument that **verifies the D5 cells**

Key Concept

"A claimed blast radius is a hypothesis until an instance has actually been killed under load."

- In the C10 contract: kill one instance/broker under load – SLO holds; blast radius $\leq$ declared – there run as a scheduled blast-radius drill (cadence *temporal*); Chaos Monkey's permanent termination is the *continual* form

DORA metrics: the delivery layer

The four DORA metrics measure whether the **delivery-relevant promises** of a structure are being kept:

Tempo	**Stability**
deployment frequency	change failure rate
lead time for changes	failed-deployment recovery time

- The central empirical finding: elite performers lead on *all four* – tempo and stability are **not a trade-off**
- In the contract: DORA four keys per deployable unit (in the cascade: *per architecture quantum*)
- Example thresholds: change failure rate $< 15\,\%$, restore $<$ 1 day; cadence continual

The coupling finding – the strongest single result in the field

(recap) deck 2, D9 and D11 already named these – today the evidence behind them

Key Concept

"Loosely coupled architectures and teams are the strongest predictor of continuous delivery."

- Supports coupling as the **leading dimension of this entire theory**
- (recap, deck 2 D9) 2017 analysis: the architecture characteristics **testability and deployability** contributed more to continuous delivery than test and deployment automation itself

Two follow-on findings matter for the matrix:

1. High performance is possible with *all kinds of systems – including mainframes* – provided systems and teams are loosely coupled; the label "microservices" is **neither necessary nor sufficient**
2. (recap, deck 2 D11 / deck 5) as team count grows, deployments per developer per day **rise** for high performers and **fall** for low performers

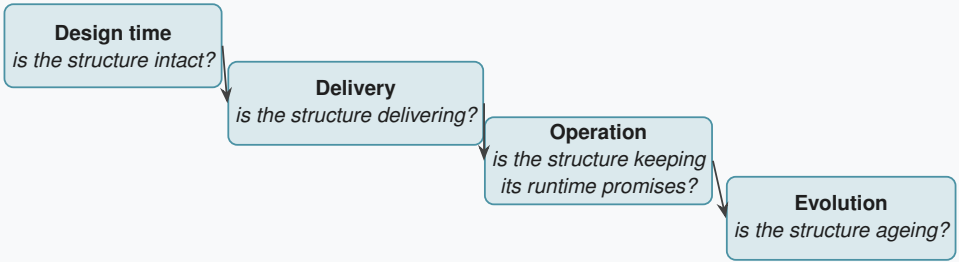
Honesty requires the caveat: prediction, not proof

Important Note

- DORA's evidence is **survey-based** and analysed with structural equation models – *prediction, not experimental causal proof*
- The theory treats it as the **best available large-*n* evidence**
- To be *triangulated* against case studies and the reader's own measurements
- Not settled law – the caveat returns in §39 as limit 4

The four-layer cascade: four falsification questions

The contract has four layers, forming a **cascade from design time to evolution**; each layer answers a different falsification question.



Instruments and example checks per layer: next frame.

The cascade: instruments and example checks per layer

Layer	Instruments	Example checks
Design time	Dependency rules as CI gates; coupling and cohesion metrics; complexity gates	“The domain layer imports no framework”; “no cycles between modules”; “no domain service calls the LLM gateway except via the declared port”
Delivery	The four DORA metrics	Deployment frequency, lead time, change failure rate, failed-deployment recovery time – *per architecture quantum*
Operation	SLOs and error budgets; latency and *cost* budgets as pipeline gates; chaos experiments as continual holistic fitness functions	p95 latency budget per scenario; token-cost budget per request; blast-radius drills
Evolution	Lehman indicators; change scatter; technical-debt inventory	Complexity trend per module; share of features touching more than two modules; debt-register review

The reference contract for C10 (1/2): structure, latency, consistency

The reference contract instantiates the cascade for the **course-project class C10** – the concrete table that ADR-007 points to.

Concern	Fitness function	Threshold (example)	Cadence
Module boundaries	ArchUnit / Spring Modulith verify: no undeclared cross-module dependency	0 violations	triggered (every build)
Determinism boundary	Static rule: no domain service imports the LLM gateway except via the declared port	0 violations	triggered
Latency	p95 end-to-end per critical scenario	advisory answer $< 20\,\mathrm{s}$	triggered + continual
Consistency	Ledger/audit reconciliation job: booked vs. journaled	0 discrepancies	temporal (daily)

Rows 1–4 of the reference contract: fitness functions with thresholds and cadences – continued on the next frame.

The reference contract for C10 (2/2): delivery, resilience, the AI rows

Concern	Fitness function	Threshold (example)	Cadence
Delivery performance	DORA four keys per deployable unit	e.g. change failure rate $< 15\,\%$; restore < 1 day	continual
Resilience	Chaos experiment: kill one instance/broker under load	SLO holds; blast radius $\leq$ declared	temporal
AI correctness	Eval-harness pass rate on golden set plus domain axioms	$\geq 95\,\%$ pass; 0 ontology-violating outputs shipped	triggered (every prompt/model change)
AI cost	Token cost per request, per feature	$\leq$ budget (e.g. CHF 0.40/request at p95)	continual

Resilience cadence: the contract runs the chaos experiment as a scheduled blast-radius drill (temporal); Chaos Monkey's permanent termination (37.1) is the continual form of the same instrument.

For AI components the contract gains **one artefact of the first rank – the eval harness**: a versioned suite of test cases, scoring logic, and statistical thresholds that runs in CI like a test suite and gates every prompt change, model update, and provider migration. Part V develops it in full.

The cost of change – what is flat

Why does the contract matter economically?

Boehm (1981 / 2001) – the classical answer

- **Boehm's cost-of-change escalation**: on waterfall project data of the 1970s, fixing a problem after delivery is up to **one hundred times** more expensive than fixing it during requirements and design
- honest qualification: for small, uncritical systems the factor is closer to **2:1**

Menzies et al. (2017) – the replication

- modern practice has empirically **flattened** that curve for changes *within* an architecture
- the largest replication to date – **171 projects from 2006–2014** – found *no consistent delayed-issue effect*

Version control, automated tests, and continuous delivery did exactly what the economic argument of Extreme Programming said they would.

The cost of change – what is still steep

- The nuance the module insists on: flattened is the curve for changes *within* an architecture (deck 1, Assumption A5 – stated then, evidenced now)
- For changes *of* the architecture – **splitting a monolith**, **changing the communication paradigm**, **moving a data-intensive flow across expensive distributed boundaries** – the curve remains steep
- The evidence is the case studies themselves: Segment's consolidation and Prime Video's rewrite were, at their core, **expensive architecture revisions**
- This asymmetry is the **economic justification of the whole apparatus**: justify the fit *up front* (architecture revision is the change class that still costs) and keep the architecture *evolvable under a green contract* (everything else is now cheap to change)

Key concept: the contract as a standing experiment

Maintenance and evolution consume roughly **40–80 %** – typically about **60 %** – of life-cycle cost, mostly for *enhancement* rather than repair (deck 1, A5 – stated then, sourced now); the contract is how a structure **earns the right to survive that phase**.

Key Concept

The measurement contract converts an architecture decision into a **standing experiment**:

- design-time gates verify the *structure*
- DORA metrics verify the *delivery*
- budgets and chaos experiments verify the *runtime promises*
- Lehman indicators verify the *ageing*

The cost-of-change curve is **flat inside a green contract** and **steep across architecture boundaries** – which is why *the contract, not the diagram*, is the artefact that protects the investment.

AI Lens (Axis A): fitness functions as the operating licence for agents

AI Lens: Axis A – fitness functions as the operating licence for agents

- An agentic coding tool iterating against a dense test suite and CI-enforced architecture rules is **contained**
- Every generated change must pass the *same* dependency rules, budgets, and evals as a human change – the blast radius of “almost right” code is bounded by the contract
- Without those gates, every agent change is **unpriced risk**
- The empirical record shows AI adoption *amplifying* existing delivery dysfunction rather than fixing it (DORA 2025 AI report)
- The measurement contract is therefore the prerequisite for raising the change rate by an order of magnitude safely: **fitness functions are the operating licence for agents**

AI Lens (Axis B): two new fitness-function types with old mechanics

AI Lens: Axis B – eval pass rate and token budget

The contract absorbs AI components through **two new fitness-function types with old mechanics**:

- **Eval-harness pass rate**: a *triggered* gate on every prompt and model change, statistically thresholded
- **Token-cost budget per request**: a *continual* gate, exactly analogous to a performance budget

Cost per request is a runtime quality attribute with **no counterpart in classical profiles**; making it a fitness function is what turns FinOps from a *monthly surprise* into an **architectural control loop**.

(deck 6, C10 profile) D9 = H in its eval reading, D10 = H cost per request – these are the two rows now measurable.

Project link: your submission ships its contract

Project Link: Portfolio Intelligence Platform

Your project submission must ship its **measurement contract**, not just its architecture – the repository must contain, *wired into CI*:

1. module-boundary verification with **zero violations** (Python: import-linter; Java: ArchUnit or Spring Modulith verify), including the determinism-boundary rule: no domain service reaches the LLM gateway except via its declared port
2. an eval harness with a versioned golden set and a **pass rate** $\geq 95\,\%$ gating every prompt or model change
3. a **token-cost budget per request** enforced as a pipeline gate and reported per feature
4. a p95 latency budget for the advisory scenario ($< 20\,\mathbf{s}$ end-to-end)
5. the ADR (in MADR form) whose final section *is* this contract

At the project review you will be asked to demonstrate **one contract violation being caught by CI** – *a contract that has never failed is a contract that has never been tested.*

The third fit dimension: why do correct matrix readings still fail?

Why do correct matrix readings still fail in real organisations?

- (recap, deck 2 D11) Conway's law in one sentence, and its consequence: every architecture decision is a team-structure decision – named there, sourced here
- The matrix matches patterns to application classes; D11 (team scaling) has appeared throughout as *one dimension among twelve* – this section makes explicit why it is more than that: the organisation is a **third party to the fit**, and ignoring it is the most common way correct matrix readings fail in practice
- The source is older than every pattern in the matrix – Conway, 1968:

Key Concept

"Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure."

- Named "Conway's law" by Brooks (1975); empirically supported by mirroring studies of organisation and product structure (MacCormack et al. 2012)

Every architecture decision is a team-structure decision

The consequence for this theory is direct: every architecture decision is *simultaneously* a team-structure decision, whether acknowledged or not.

MS to one team

- **a microservices topology assigned to a single five-person team** produces a **distributed monolith**
- many quanta, one communication structure, the worst cells of *two* columns at once

(recap, deck 5: the distributed monolith and its lockstep-release signature)

MM to thirty teams

- **a modular monolith assigned to thirty independent teams** produces a **release-coordination bottleneck**
- that no amount of code quality repairs

Team Topologies: four team types, three interaction modes

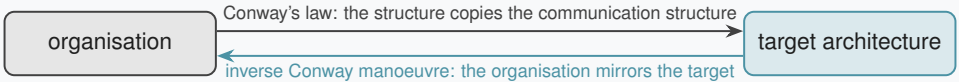
Team Topologies (Skelton and Pais 2019) turns the law **from a hazard into a design instrument**. Its four fundamental team types:

Team type	Role
Stream-aligned	delivering end-to-end on one value stream
Platform	reduce the load of stream teams
Enabling	build missing capabilities
Complicated-subsystem	encapsulate specialist knowledge – *an ML inference subsystem is the canonical course-relevant example*

- Three interaction modes: **collaboration**, **X-as-a-service**, **facilitating**
- Together: the vocabulary for matching team structure to pattern choice

Two concepts that bind directly into the matrix

1. **Cognitive load as a design criterion**
   - team and software boundaries should be cut so that *no team's cognitive load exceeds its capacity*
   - "team-sized software" is an **architectural yardstick**
   - it explains why the MS column demands stream-aligned teams with full ownership as a **precondition, not an outcome** (recap, decks 5 and 6: stream-aligned + platform – a precondition)
2. **The inverse Conway manoeuvre**
   - deliberately structure the organisation to mirror the *target* architecture
   - so that Conway's law works *for* the design instead of against it



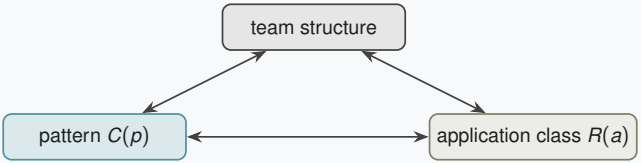
The empirical anchor – and the contested rows D11 decides

- The DORA scaling finding, read again: loose coupling of *architectures and teams* – measured **jointly**, which is itself a Conway statement – is the strongest predictor of continuous delivery
- It is the mechanism that lets deployments per developer per day **scale linearly with team count**
- The matrix encodes the organisational variable in D11; the class rationales repeatedly show it *deciding contested rows*:

Class	Same profile	Smaller organisation	Larger organisation
C1	one C1 profile, different organisation sizes	LMAX: single-threaded core	Monzo: 2,800 services
C5	one C5 profile, split along team count, not traffic	Shopify: modular monolith	Amazon: microservices

- No twelve-dimensional profile fully captures an organisation (limit 5 in §39) – but the rule of thumb is teachable

Key concept: the fit is three-way



Key Concept

The fit is three-way: **pattern** $\leftrightarrow$ **application class** $\leftrightarrow$ **team structure**. **Check D11 last but veto on it first**: a pattern whose team precondition is not met – microservices without stream-aligned ownership, a monolith across too many coordinating teams – will fail regardless of how well the other eleven dimensions match. If the target architecture and the organisation disagree: either apply the **inverse Conway manoeuvre** or **change the target**. *Conway's law does not negotiate.*

Limit 1: ordinal scales, no arithmetic

A theory whose declared standard is that unfalsifiable claims have no place in architecture decisions must state how it can itself mislead. Six limits, stated plainly – the first:

- The ratings support **rankings and exclusions, never percentages**
- Any weighted-sum reading of the matrix imports the documented defects of additive multi-criteria methods over ordinal data:

rank reversal Belton and Gear 1983	**axiomatic conflict with utility theory** Dyer 1990
scale misinterpretation Bana e Costa and Vansnick 2008	**pseudo-precision** priorities with three decimal places from coarse verbal comparisons

- We keep the *explication discipline* of multi-criteria decision analysis and drop its *arithmetic pretensions*
- **Sensitivity analysis is mandatory, not decorative**; unstable rankings are *findings* (tradeoff points), not errors

Limits 2 and 3: context-dependent cells, hybrids as the normal case

Limit 2. Context-dependence of ratings – every capability cell encodes a *typical* workload

- the serverless cost cell literally inverts with load shape (Prime Video); the layered read-scalability cell inverts with cache-friendliness (Stack Overflow)
- skilled teams can move individual cells – LMAX and Monzo both did
- a rating is a **default hypothesis** to be replaced by measurement in step (viii); *the matrix predicts the default, not the exceptional*

Limit 3. Hybrids are the normal case, not the exception

- eight of ten class recommendations involve a core pattern plus different edge patterns
- the matrix is defined over **subsystems**; applying it to a whole enterprise in one stroke is a *category error the theory explicitly forbids*

Limit 4: the evidence base is heterogeneous

Source	Nature	Weakness
Star ratings (Richards and Ford)	expert judgement	not measurements
DORA	survey-based prediction	not causal proof
Case studies	self-reported engineering blogs	selection and framing bias
Hexagonal and serverless	–	no star ratings at all

Prime Video in particular is routinely misquoted as “Amazon abandons microservices” when it documents *one service* with a data-intensive streaming workload – you met it in deck 6 (SL, the split cell).

Compensation: **triangulation** – ratings against cases against metrics – and the **measurement contract**, which converts every adopted claim into a testable one.

Limits 5 and 6: the hidden organisation, AI-era volatility

Limit 5. The organisation is a hidden variable – Conway's law makes every architecture decision a team-structure decision

- the same requirements profile admits *opposite* optimal patterns at different organisation sizes (LMAX vs. Monzo; Shopify vs. Amazon)
- D11 partially captures this; no twelve-dimensional profile fully does (§38)

Limit 6. AI-era volatility – the D12 ratings encode the **2025/26 state** of a field whose tools deprecate in months

- the *method* – Assumption A6: scenarios, tactics, trade-off analysis, ADRs, fitness functions – is the stable part; the specific cells are **perishable**
- they carry, in effect, their own **temporal fitness function**: re-verify on every model generation – *Lehman's laws apply to theories too*

Six limits – and what compensates each

Limit	What it means	What compensates
1 Ordinal scales	rankings and exclusions only	sensitivity analysis, mandatory; unstable rankings are findings
2 Context-dependence	cells are typical-workload defaults	measurement replaces the rating in step (viii)
3 Hybrids normal	8 of 10 classes core + edges	apply the matrix to subsystems, never to a whole enterprise
4 Heterogeneous evidence	expert stars, survey prediction, self-reported blogs	triangulation + the measurement contract
5 Hidden organisation	same profile, opposite optima by organisation size	D11 captures it only partially – check the team precondition separately (§38)
6 AI-era volatility	D12 cells encode 2025/26	method (A6) stable; cells re-verified on every model generation

Important note: the matrix is a hypothesis, not an authority

Important Note

- A student who cites the matrix as an *authority* rather than as a *hypothesis* has misunderstood the module
- The matrix cannot tell you what to build; it can only **force your criteria, weights, and assumptions into the open**, **pre-filter the candidates**, and **hand the contested cells to scenario-based analysis**
- Its numbers are ordinal, its ratings are context-typical defaults, its evidence is triangulated but partly survey-based and partly self-reported – and it *decays*: every cell is a claim awaiting your measurement
- The canonical exercise in this scepticism: reading contradictory study designs against each other – the METR-versus-Copilot contradiction of Part V (next week)
- The assessment of this module tests the **discipline**, not the memorisation of the grid

Key concept: Maxim 9

Key Concept

Maxim 9. The matrix is the argument's **skeleton**, ATAM is its **court of appeal**, the ADR is its **record**, and the fitness function is its **parole condition**.

The matrix is the *lecture-hall form* of a discipline whose *engineering form* is:

scenarios with numbers · ATAM for the contested cells · ADRs for the decisions · fitness functions for the lifetime

Discussion

Discussion

Limit 4 says the evidence base is heterogeneous, and limit 2 says skilled teams can move cells. Suppose your team measures, over a year, that its microservices system beats every prediction of the MS column for its class.

- Has the theory been **falsified** – or has your measurement contract done exactly what step (viii) designed it to do?
- What would have to be true of your *next* project for the difference to matter?

This week's exercise: resilience complete – M4 closes

Project Link: Portfolio Intelligence Platform

- Coaching session (1 lesson). **Resilience patterns on all external calls**: timeout, retry, circuit breaker, fallback
- **Graceful degradation verified**
- **Milestone M4 of the exercise sheet closes (end of week 11)**: deterministic core *fully tested* against the reference vectors *and resilient*
- Hint: keep the deterministic core free of LLM calls – *this is the line that is graded*

Looking ahead: the resilience row of the reference contract (kill one instance/broker under load – SLO holds) is the fitness function that would verify what you build now – it is not a graded deliverable.

Summary

1. **Fitness functions**: objective integrity assessments; scope atomic/holistic, cadence triggered/continual/temporal; three families – dependency gates, budgets, chaos experiments
2. **DORA**: four keys, elite performers lead on all four; loosely coupled architectures *and teams* predict continuous delivery – prediction, not causal proof
3. **Four-layer cascade** – four falsification questions; the C10 reference contract, eight rows with thresholds and cadences
4. **Cost of change**: flat *within*, steep *across* architecture boundaries – the contract, not the diagram, protects the investment
5. **AI lens**: fitness functions are the operating licence for agents (A); eval pass rate and token budget, two new types with old mechanics (B)
6. **Conway**: the fit is three-way; check D11 last but veto on it first; inverse Conway manoeuvre
7. **Six limits**: ordinal, context-dependent, hybrids, heterogeneous evidence, hidden organisation, AI-era volatility – a hypothesis, not an authority
8. **Maxim 9**: skeleton, court of appeal, record, parole condition – Part IV closes

Next week

Lecture 12 – Part V: two axes, one method

- Axis A: Copilot vs. METR – two contradictory RCTs and their resolution; the verification bottleneck
- Axis A compact: architecture documentation as control interface, guardrails, the tool landscape and MCP, risks and accountability (sections 41.6–41.9)
- Axis B (I): the news-sentiment call wired the obvious way vs. the right way; the three component types; the SE4AI classics
- Axis B (I): integration patterns – reference architecture with the LLM gateway; eval-harness foundations

Reading

- this week: Part IV, sections 37–39
- ahead: Part V, sections 40–41, 42.1–42.5

Exercise / deliverable

- coaching: AdvisorAgent + 2–3 sub-agents behind the gateway (mandatory)
- ontology guard active on all insights



Thank you!

Dr. Florian Herzog

Fachhochschule Graubünden, Chur

AISE502 – AI in Software Engineering II