AEA by Ivory & Co.

Ecosystem Architecture Standard

Version: v1.0.0 — Published Anchor: Interconnectedness

1. Definition

Ecosystem Architecture defines the actors, relationships, exchanges and dependencies through which the enterprise creates, accesses, exchanges and sustains value.

Ecosystem Architecture is the architecture of Interconnectedness.

Its governing question is:

What Actors and Relationships must exist around the enterprise, what passes between them, and what Dependencies arise through that interconnectedness?

Ecosystem Architecture establishes the structural relationship network within which the enterprise participates.

It is concerned with the enterprise as a participant in a network rather than as an isolated entity.

2. Why Ecosystem Architecture Exists

No enterprise creates, accesses, exchanges or sustains value alone.

Customers, suppliers, partners, regulators, distributors, service providers, platforms, portfolio entities, communities and other parties may all become architecturally material because the enterprise depends upon, exchanges with or participates in relationships with them.

Those relationships are not merely a stakeholder list.

They have structure.

They establish:

- who participates;
- how parties are related;
- what is exchanged;
- what each party depends upon;
- what commitments shape the relationship;
- what second-order consequences can propagate through it;
- and how material that interconnectedness is to the enterprise.

Ecosystem Architecture makes that network explicit.

Ecosystem Architecture establishes the relational structure through which enterprise value is connected to other parties.

3. Core Model

The core Ecosystem proposition is:

Actor A !” Relationship !” Actor B

Within that Relationship:

Actor A !’ Exchange !’ Actor B

Actor B !’ Exchange !’ Actor A

and:

Actor A !’ Dependency !’ Actor B

Actor B !’ Dependency !’ Actor A

where those Exchanges and Dependencies exist.

The architecture therefore follows a simple governing rule:

One Relationship. Two Actor perspectives. Directional Exchanges. Directional Dependencies.

Reciprocity must be represented where it exists.

It must not be assumed.

4. Actor

An Ecosystem Actor is a functional or instantiated entity required within the relationship architecture through which a business capability is delivered, accessed, exchanged or sustained.

An Actor exists in Ecosystem Architecture because it is material to interconnectedness.

An Actor may represent an external organisation, internal entity, customer, supplier, partner, regulator, platform, community, portfolio company, provider or another party where a material Relationship, Exchange or Dependency exists.

Actor identity may be represented functionally before the specific entity fulfilling it is known.

This enables the architecture to state:

The enterprise requires an Actor of this kind in this relationship

without prematurely deciding the specific person, company or participant that will fulfil it.

Define the Actor required before determining who or what fulfils it.

5. Actor Materiality

Not every person, organisation, system or party known to the enterprise is an Ecosystem Actor.

An Actor is architecturally material where its participation is required to understand how enterprise value is delivered, accessed, exchanged or sustained.

The governing test is:

If this Actor were removed from the architecture, would a material Relationship, Exchange, Dependency or capability-delivery condition become invisible or incorrect?

If yes, the Actor is likely material.

If no, the entity may exist operationally without requiring separate representation in Ecosystem Architecture.

Ecosystem Architecture must not become a contact directory, CRM view, organisation chart or generic stakeholder inventory.

6. Relationship

A Relationship is the structural connection between two Ecosystem Actors through which one or more Exchanges, Dependencies, commitments or other material relational conditions exist.

A Relationship is represented once as the connection between the Actors.

The two Actors may experience that Relationship differently.

Their Exchanges may differ.

Their Dependencies may differ.

Their obligations may differ.

Their materiality may differ.

Their Trust may differ.

That does not require two Relationships to be created merely to represent two perspectives.

> One Relationship can carry materially different propositions from each Actor perspective.

7. Exchange

An Exchange is something of architectural significance that passes from one Actor to another through or because of a Relationship.

An Exchange is directional.

Possible Exchanges may include:

- money;
- goods;
- services;
- information;
- data;
- knowledge;
- access;
- rights;
- obligations;
- commitments;
- referrals;
- influence;
- attention;
- capability access;
- or another material item or proposition.

This is not a closed taxonomy.

The existence of a Relationship does not require equivalent Exchange in both directions.

A regulator may receive information while providing authority or permission through a different mechanism.

A customer may provide money and receive a service.

A partner may provide access while receiving another form of value.

> Exchange symmetry must not be assumed.

8. Dependency

A Dependency is a condition in which an Actor's ability to create, access, exchange or sustain value depends materially upon another Actor, Relationship, Exchange or relational condition.

Dependencies are directional.

Actor A may depend heavily upon Actor B while Actor B has little or no corresponding Dependency upon Actor A.

Dependencies may concern:

- supply;
- access;
- demand;
- capability;
- information;
- timing;
- approval;
- continuity;
- infrastructure;
- distribution;
- relationship access;
- contractual performance;
- or another material condition.

A Dependency describes structural interconnectedness.

It does not determine whether reliance is warranted.

> Dependency may create exposure to reliance. Dependency is not itself warranted Reliance.

Trust Architecture determines whether, under what conditions and to what extent one party can rely upon another.

9. Relationship Instruments

A Relationship may be established, bounded or evidenced through a Relationship Instrument.

Examples may include:

- contract;
- agreement;
- memorandum;
- licence;
- partnership instrument;
- service agreement;
- regulatory instrument;
- terms of participation;
- or another formal or informal mechanism that establishes material relational commitments.

The Relationship Instrument is relevant to Ecosystem because it establishes or qualifies the relationship and its commitments.

It does not make actual performance an Ecosystem proposition.

The cross-architecture sequence is:

> Ecosystem establishes the Relationship and its commitments !’ Reality captures what actually occurs against them !’ Trust determines warranted Reliance.

A contractual promise is not evidence that the promise was fulfilled.

Actual fulfilment is a Reality proposition.

Whether another party can consequently rely upon the party is a Trust proposition.

10. Relationship Commitments

A Relationship Commitment is a material obligation, promise, entitlement, constraint or expectation established through the Relationship or its Instrument.

Ecosystem Architecture may preserve these commitments because they define the structural terms under which Actors participate.

Execution may operationalise work required to satisfy them.

Reality may capture what actually occurs against them.

Trust may determine the reliance consequence of consistency or inconsistency against them.

> The commitment belongs to the Relationship. Its performance is Actuality. Its reliance consequence is Trust.

11. Relationship Materiality

Relationship Materiality expresses the architectural significance of a Relationship to the enterprise.

Materiality may arise because the Relationship materially affects:

- capability delivery;
- access to value;
- exchange;
- continuity;
- strategic position;
- concentration;
- dependency;
- route to market;
- regulatory permission;
- supply;
- demand;
- ecosystem access;
- or another architecturally significant condition.

Relationship Materiality is not a universal numerical score.

The detailed assessment method belongs in the downstream Ecosystem Architecture package.

The Published Standard establishes the proposition:

> Not every Relationship deserves equal architectural resolution. Materiality determines where greater relational detail is required.

12. Second-Order Relationship Impact

A Second-Order Relationship Impact is a material consequence for an Actor arising from a Relationship or Relationship Instrument to which that Actor is not a direct party.

For example:

> Actor A !” Actor B !” Actor C

A commitment between B and C may materially affect A even though A is not party to that

Relationship.

Second-order analysis allows Ecosystem Architecture to reveal consequences that would be invisible in a purely pairwise stakeholder map.

Traversal should continue only while an external Relationship or Relationship Instrument remains materially relevant to the enterprise's Execution, value delivery or success measures.

Ecosystem Architecture does not require an exhaustive map of every relationship in every Actor's wider world.

> Second-order traversal is governed by material consequence, not theoretical connectedness.

13. Network Topology

Ecosystem Architecture is a network architecture.

It must not assume that every relationship is hub-and-spoke around the enterprise.

Actors may have material Relationships with one another that affect enterprise value even where the enterprise is not a direct party.

The architecture may therefore represent:

- direct Relationships;
- second-order Relationships;
- clusters;
- platform-mediated Relationships;
- supply networks;
- distribution networks;
- regulatory networks;
- portfolio networks;
- and other material topologies.

The enterprise remains the architectural scope of concern, but not necessarily the centre of every Relationship.

14. Capability-Driven Actor Completeness

Domain Architecture provides a primary completeness test for Ecosystem Architecture.

For every material Capability in Domain Architecture, ask:

> What Actors and Relationships are required for that Capability to be delivered, accessed, exchanged or sustained?

The reverse test also applies:

> For every Ecosystem Actor, which Capability or Capabilities establish why that Actor is architecturally material?

These tests create traceability between stable enterprise ability and the relationship network required around it.

They do not require one Actor per Capability.

One Actor may support many Capabilities.

One Capability may require many Actors.

Some Capabilities may be predominantly internal and require minimal Ecosystem

representation.

Domain establishes why capability must exist. Ecosystem establishes the Actor and Relationship architecture required around its delivery, access, exchange or sustainment.

15. Actor Before Fulfilment

Ecosystem Actor identity is deployment-agnostic.

The architecture first determines what relational participant is required.

Only then should the enterprise determine who or what fulfils that requirement where fulfilment is relevant.

This preserves the distinction between relationship architecture and workforce composition.

The governing sequence is:

Domain — what Capability must exist

Ecosystem — what Actors and Relationships are required around its delivery, access, exchange or sustainment

Execution — what work must be performed

Workforce — what Roles are required and who or what fulfils them

An Ecosystem Actor may later be instantiated by a particular organisation or other entity.

That does not make the Workforce Role and Ecosystem Actor the same architectural object.

16. Actor, Role and Participant

Actor "` Role "` Participant.

An Actor is an Ecosystem proposition.

A Role is a Workforce proposition composed from participation requirements established through work.

A Participant is the human, agent or organisation fulfilling a Workforce Role or participation arrangement.

The same real-world entity may appear across several architectures without those objects becoming interchangeable.

For example, a supplier organisation may be:

· an Ecosystem Actor because it participates in a supplier Relationship;
· a Workforce Participant where the enterprise has sourced an Execution Outcome or other participation from that organisation;
· an Expression Audience where contextual expression to that supplier is material.

These are different architectural readings of the same entity.

Shared identity does not create shared object ownership.

17. External Provider Boundary

Where the enterprise sources a Capability or Execution Outcome from another organisation, that organisation may be an Ecosystem Actor because the enterprise holds a material Relationship with the provider.

The provider's internal workforce does not automatically become part of the enterprise's

Ecosystem Architecture.

Nor does it automatically become part of the enterprise's Workforce Architecture.

The architectural boundary follows what the enterprise actually owns and needs to represent.

Where the enterprise buys or partners for an outcome or Capability, the provider's internal composition may remain opaque.

Where the enterprise directly sources Labour or Skill against its own Roles, Workforce Architecture may require participant-level representation.

> A relationship with a provider does not automatically make every participant inside that provider an Ecosystem Actor.

18. Ecosystem and Workforce Structures

Ecosystem Architecture establishes the relationship network in which Actors participate.

Workforce Architecture establishes four orthogonal structures through which Roles and Participants are arranged:

1. Delivery Structure — how Roles and Participants are arranged to perform work. 2. Organisation Structure — formal organisational accountability and placement. 3. Authority Structure — binding decision rights and delegation for execution. 4. Discipline Structure — professional, technical or practice standards and oversight.

None of these structures is an Ecosystem structure.

A Workforce structure may intersect with an Ecosystem Relationship without becoming that Relationship.

For example:

- an external professional body may be an Ecosystem Actor;
- the relationship with that body may establish a qualification or professional obligation;
- Workforce Discipline Structure may use that standard to govern competent practice;
- Workforce Authority Structure separately determines binding decision rights within the enterprise.

> External relationship does not equal internal structure.

19. Discipline, Standards and Authority

An external regulator, professional body, certification body or standards organisation may be an Ecosystem Actor where the enterprise participates in a material Relationship with it.

The external body may establish obligations, credentials or standards through that Relationship.

Workforce Discipline Structure governs how competent practice is organised internally.

Workforce Authority Structure governs binding decision rights and delegation.

Trust may use the standing of the external body, credential or authority as a Reliance Condition.

These propositions must remain separate.

> A standard may shape practice without conferring enterprise authority.

> A relationship may establish an obligation without becoming the internal Authority Structure.

20. Ecosystem and Trust Architecture

Ecosystem Architecture establishes:

- the Actors;
- the Relationship;
- the Exchanges;
- the Dependencies;
- and relevant relational commitments.

Trust Architecture determines warranted Reliance between parties.

The distinction is fundamental.

> The existence of a Relationship does not establish Trust.
>
> The existence of a Dependency does not establish warranted Reliance.
>
> The existence of a contract does not prove performance.

Trust may determine that Party A can rely upon Party B for a particular Reliance Attribute, under defined Reliance Conditions, to a particular extent and on a particular Basis of Reliance.

That proposition remains Trust-owned.

Ecosystem Architecture preserves the structural Relationship in which that Trust proposition exists.

Trust may be asymmetric even where the Relationship is mutual.

Trust may change materially while the Ecosystem Relationship remains structurally unchanged.

This is why Trust is not a property of the Ecosystem Relationship.

21. Transferred Trust and Network Propagation

Transferred Trust may traverse Ecosystem Relationships.

For example:

> Actor A relies upon Actor B !’ Actor B vouches for Actor C !’ Actor A extends initial Reliance to Actor C

Ecosystem Architecture provides the network topology through which that propagation can be understood.

Trust Architecture owns:

- the Basis of Reliance;
- Transferred Trust;
- Trust Scores;
- Trust Impact;
- and Trust Blast Radius.

Ecosystem does not acquire those propositions because the Trust relationship traverses its network.

> Ecosystem shows the network through which reliance may propagate. Trust determines the reliance.

22. Ecosystem and Reality Architecture

Ecosystem establishes the structural Relationship and its commitments.

Reality captures what is actually occurring.

This may include actual:

- exchange volume;
- delivery;
- non-delivery;
- relationship events;
- dependency conditions;
- contract performance;
- concentration;
- interaction;
- or another material actuality.

Reality does not acquire ownership of the Actor or Relationship because it captures their actual state.

Ecosystem does not own actual performance merely because performance concerns a Relationship.

> Ecosystem establishes. Reality captures correspondence.

Trust may then use relevant Actuality when determining warranted Reliance.

23. Ecosystem and Intelligence Architecture

Ecosystem originates Requirements of Understanding where it needs to understand an Actor, Relationship, Exchange, Dependency, commitment, second-order impact or another Ecosystem proposition.

Intelligence architects fulfilment of those requirements.

Information about an Actor is not the Actor.

Information about a Relationship is not the Relationship.

Intelligence may establish understanding of the network without acquiring ownership of it.

> Ecosystem determines what it needs to understand. Intelligence establishes how that understanding is formed.

24. Ecosystem and Expression Architecture

Ecosystem establishes the Actor universe relevant to the enterprise's relationship architecture.

Expression determines how those parties should be treated as audiences for contextual expression.

> Actor Type "` Audience Segment.

One Actor Type may require multiple Audience Segments.

Several Actor Types may share one Audience Segment.

An Actor may require no distinct Expression segmentation.

The governing boundary is:

> Ecosystem establishes the Actors with whom the enterprise participates in relationships. Expression determines the audience segmentation required for the contextual expression of Enterprise Identity.

An Exchange may carry an Expression.

A Relationship may provide the context in which an Expression occurs.

Neither fact transfers Actor or Relationship ownership to Expression.

25. Ecosystem and Directional Architecture

Directional owns strategic scope and choices.

This includes strategic Route-to-Market choices.

Where Directional establishes that the enterprise will participate through a particular channel, market pathway, partnership model or route to market, Ecosystem establishes the Actors, Relationships, Exchanges and Dependencies required to make that choice structurally possible.

> Directional chooses the route. Ecosystem establishes the relationship network through which the route operates.

Ecosystem does not independently make the strategic choice merely because it exposes the relationships required to support it.

26. Ecosystem and Domain Architecture

Domain owns Business Models, Value Chains and Capabilities.

Ecosystem owns the Actors and Relationships required around the delivery, access, exchange and sustainment of those Capabilities.

The relationship is many-to-many.

A Capability may depend upon several Actors.

An Actor may participate around several Capabilities.

Ecosystem must not redefine Capability identity merely because an Actor provides or participates in it.

Domain must not absorb the Actor network merely because a Capability depends upon it.

27. Ecosystem and Execution Architecture

Ecosystem establishes relational commitments, Exchanges and Dependencies.

Execution owns the instructions through which associated work is performed.

A contract may require an outcome.

Ecosystem preserves the relationship commitment.

Execution may establish the Process Module, Variant, Task, Instruction, Operational Policy or RACI required to perform the work.

> Relationship obligation is not the same proposition as the instruction used to satisfy it.

Where a Relationship changes but executable work does not, no Execution change is necessarily required.

Where the Relationship change materially alters executable work, Execution must assess the impact.

28. Ecosystem and Environment Architecture

An external provider may be an Ecosystem Actor.

The physical or technological enablement provided through that Relationship may be an Environment proposition.

For example, a cloud provider may be represented in Ecosystem as an Actor with a contractual Relationship and material Dependency.

The cloud environment, platform or technological enablement used by the enterprise remains Environment Architecture.

Ecosystem owns the provider Relationship. Environment owns the physical or technological enablement requirement and fulfilment.

Actual availability or performance remains Reality.

29. Ecosystem and Possibility Architecture

Ecosystem Architecture represents established or architecturally required interconnectedness.

Possibility Architecture explores what could become possible.

A potential partner, possible channel, prospective alliance or hypothetical dependency may be explored through Possibility before becoming established Ecosystem Architecture.

Likewise, an existing Ecosystem condition may create, constrain, amplify or compound future optionality.

Possibility owns that future option proposition.

Ecosystem owns the established relationship structure.

Ecosystem describes the relationship network that is or is required. Possibility explores how that network could change or what it could make possible.

30. Route to Market

Route to Market spans multiple architectures.

The governing distinction is:

Directional — which route the enterprise strategically chooses

Ecosystem — which Actors, Relationships, Exchanges and Dependencies make that route operate

Expression — how Enterprise Identity and value are expressed through that route

Execution — what work is performed through it

Environment — what physical and technological enablement supports it

The word “channel” or “route” must not be assigned wholesale to one architecture.

The proposition determines ownership.

31. Ecosystem Architecture as a Network, Not a Stakeholder Map

A stakeholder list tells the enterprise who may matter.

Ecosystem Architecture establishes why they matter structurally.

A conformant Ecosystem Architecture must be capable of representing more than names or categories.

It must reveal the network of:

Actors !’ Relationships !’ Exchanges !’ Dependencies

and the directional structure between them.

It may also preserve Relationship Instruments, commitments, materiality and second-order impacts where architecturally material.

A stakeholder map may be a useful input or downstream view.

It is not, by itself, Ecosystem Architecture.

32. Governing Principles

Ecosystem Architecture is the architecture of Interconnectedness.

Ecosystem Architecture defines the actors, relationships, exchanges and dependencies through which the enterprise creates, accesses, exchanges and sustains value.

One Relationship. Two Actor perspectives. Directional Exchanges. Directional Dependencies.

Reciprocity must be represented where it exists, not assumed.

Define the Actor required before determining who or what fulfils it.

Actor materiality is grounded in the enterprise's value and Capability architecture.

For every material Capability, test what Actors and Relationships are required around its delivery, access, exchange or sustainment.

For every Ecosystem Actor, test which Capability or Capabilities establish why that Actor is architecturally material.

Dependency may create exposure to reliance. Dependency is not itself warranted Reliance.

Relationship existence does not establish Trust.

A Relationship Instrument establishes or qualifies commitments. It does not prove actual performance.

Ecosystem establishes the Relationship and commitments. Reality captures what actually occurs. Trust determines warranted Reliance.

Second-order traversal is governed by material consequence, not theoretical connectedness.

Actor "` Role "` Participant.

Actor Type "` Audience Segment.

Shared real-world identity does not create shared architectural ownership.

A relationship with a provider does not automatically make the provider's internal workforce part of the enterprise's Ecosystem or Workforce Architecture.

Workforce Delivery, Organisation, Authority and Discipline Structures are not Ecosystem structures.

Ecosystem shows the network through which reliance may propagate. Trust determines the reliance.

Directional chooses the Route to Market. Ecosystem establishes the relationship network through which it operates. Expression determines contextual expression through it.

Ecosystem Architecture is not a stakeholder list, contact directory, org chart, process map or Trust model.

33. AI Protocol

AI may assist Ecosystem Architecture by:

· identifying candidate Actors from established Capability and relationship evidence;
· identifying candidate Relationships;
· extracting Exchanges and their direction;
· identifying Dependencies and their direction;
· detecting missing reciprocity or falsely assumed reciprocity;
· identifying Relationship Instruments and commitments;
· identifying potential second-order Relationship impacts;
· testing Actor materiality against Domain Capabilities;
· identifying concentration and dependency patterns;
· testing whether an Actor has been incorrectly represented as a Role or Participant;
· testing whether an Actor Type has been incorrectly treated as an Audience Segment;
· and tracing where Trust propositions should be routed to Trust Architecture.

AI must not:

· create an Actor merely because a person or organisation appears in a source;
· create an Ecosystem Actor from every named stakeholder;
· assume reciprocity;
· infer an Exchange where none is established;
· infer a Dependency merely because two Actors are connected;
· treat Dependency as warranted Reliance;
· infer Trust from the existence of a Relationship;
· infer actual performance from a contractual commitment;
· treat an Actor as a Workforce Role;
· treat a Workforce Participant as an Ecosystem Actor without a material relationship basis;
· treat Actor Type as Audience Segment;
· invent Audience Segments;
· make a strategic Route-to-Market choice;
· turn an external standard-setter into an internal Authority Structure;
· map every relationship of every Actor merely because network traversal is technically possible;
· or expose proprietary AEA in Action composition methods as Published Standard doctrine.

AI may reveal interconnectedness. It must not manufacture relationships, reliance, reciprocity or materiality.

34. Conformance

A conformant application of Ecosystem Architecture must preserve, where applicable:

1. the relevant Actor; 2. Actor materiality; 3. the Relationship; 4. both Actor perspectives where material; 5. directional Exchanges; 6. directional Dependencies; 7. Relationship Instruments where material; 8. material Relationship Commitments; 9. Relationship Materiality where required; 10. Second-Order Relationship Impact where material; 11. capability traceability sufficient to explain Actor materiality; 12. distinction between Actor and Workforce Role; 13. distinction between Actor and Workforce Participant; 14. distinction between Actor Type and Expression Audience Segment; 15. distinction between Dependency and warranted Reliance; 16. distinction between Relationship commitment and actual performance; 17. distinction between Ecosystem Relationship and Workforce structures; 18. distinction between strategic Route-to-Market choice and Ecosystem relationship structure; 19. Requirements of Understanding originating in Ecosystem where applicable; 20. and cross-architecture ownership where another architecture consumes Ecosystem information.

35. Non-Conformance

An application is non-conformant where it:

· reduces Ecosystem Architecture to a stakeholder list;
· reduces Ecosystem Architecture to an external environment scan;
· represents named people without an architectural Actor basis;
· assumes every Relationship is reciprocal;
· assumes Exchanges are symmetrical;
· assumes Dependencies are symmetrical;
· creates duplicate Relationships merely to represent each Actor perspective;
· treats Dependency as Trust;
· treats the existence of a Relationship as proof of warranted Reliance;
· treats a contractual commitment as proof of actual performance;
· absorbs Trust Scores or Trust state into the Ecosystem Relationship;
· treats Workforce Role as an Ecosystem Actor;
· treats Workforce Participant as an Ecosystem Actor without a material Relationship basis;
· treats Actor Type as Audience Segment;
· makes Workforce Delivery, Organisation, Authority or Discipline Structure an Ecosystem object;
· treats a regulator or standards body Relationship as internal enterprise authority;
· treats the provider's internal workforce as enterprise Actors merely because the enterprise contracts with the provider;
· treats Route-to-Market strategic choice as Ecosystem-owned;
· or traverses network relationships without a material enterprise consequence.

36. Published Gaps

GAPS — None currently identified that prevent publication of this Standard.

Detailed Actor fields, Relationship classifications, Exchange classifications, Dependency attributes, Relationship Materiality methods, Relationship Instrument fields, second-order traversal protocols and native schematic construction rules belong in the downstream Ecosystem Architecture package.

Their absence from this Published Standard does not constitute an architecture doctrine gap.

37. Publication Position

Ecosystem Architecture provides AEA with its architecture of Interconnectedness.

It establishes the structural network through which the enterprise participates in value creation, access, exchange and sustainment.

Its native logic is relational:

Actor A !” Relationship !” Actor B

with:

directional Exchanges

and:

directional Dependencies.

It preserves the difference between the existence of interconnectedness and the consequences that other architectures determine from it.

A Dependency does not automatically establish Trust.

A Relationship Commitment does not establish Actuality.

An Actor does not become a Workforce Role.

An Actor Type does not become an Audience Segment.

A provider Relationship does not make the provider's internal workforce part of the enterprise.

And a strategic Route-to-Market choice does not become Ecosystem-owned merely because Ecosystem makes that route structurally possible.

The architecture therefore provides the relationship skeleton upon which several other architectural propositions may depend without absorbing those propositions into itself.

The governing model is:

Domain establishes the Capabilities that make Actor participation material.

Ecosystem establishes the Actors, Relationships, Exchanges and Dependencies around that value.

Execution establishes the work required.

Workforce establishes the Roles and Participants that fulfil participation in that work.

Reality captures what actually occurs.

Trust determines warranted Reliance.

Expression determines contextual expression to relevant audiences.

And throughout:

Cross-architecture contribution does not transfer ownership.