ArchiMate Relationship Matrix

Architect Edition

Layer map • Full relationship matrix • Semantic guidance • Cross-layer rules • 10 anti-patterns

Nilus Consulting — 2026 Edition • ArchiMate 3.2 Compliant

Strategy	Resource, Capability, Value Stream, Course of Action
Business	Actor, Role, Process, Function, Service, Event, Object, Contract, Product
Application	Component, Interface, Function, Process, Service, Event, Data Object
Technology	Node, Device, System SW, Network, Path, Function, Service, Artifact
Physical	Equipment, Facility, Distribution Network, Material

Motivation

Implementation & Migration

Business Layer — Intra-Layer Relationship Matrix

	Actor	Role	Collab	IFace	Process	Function	Event	Service	Object	Contract	Product
Actor	SP	A	AG						AC		
Role		SP	AG	C	A	A			AC		
Collab			SP	C	A	A			AC		
Interface				SP				S	AC		
Process					SP		T	R	AC		
Function						SP	T	R	AC		
Event					T	T	SP		AC		
Service								SP	AC		AG
Object									SP	R	R
Product								AG		AG	SP

Cross-Layer Relationships

	Bus Proc	Bus Func	Bus Svc	App Comp	App Svc	App IFace	Tech Node	Tech Svc
App Component			R	SP	C	C		
App Service	S	S	S		SP			
App Function			R		R			
Tech Node				A			SP	
Tech Service				S	S			SP
Artifact				R				

Relationship Semantics & Top 10 Anti-Patterns

Assignment

Meaning: Who performs this behavior?

Direction: Active → Behavioral (intra-layer ONLY)

Example: Role → Process

Mistake: Using across layers where Realization is needed

Realization

Meaning: Implements an abstract element

Direction: Concrete → Abstract (CROSS-layer bridge)

Example: App Component → Bus Service

Mistake: Realizing same-type elements (Svc → Svc)

Serving

Meaning: Provides functionality to another

Direction: Provider → Consumer (lower → upper)

Example: App Service → Bus Process

Mistake: Serving downward (Business → Technology)

Top 10 ArchiMate Modeling Anti-Patterns

1. Overusing Association
Use specific relationships instead of generic

2. Using Flow for everything
Flow = value/info transfer. Triggering = causal order

3. Skipping Realization
Always show how lower layers realize upper services

4. Business-to-Technology shortcuts
Always model the Application layer in between

5. Misusing Access as dependency
Access = read/write data. Serving = service dependency

6. Data Objects as processes
Data Objects are passive. They do not trigger or serve

7. Over-nesting Compositions
Keep composition to 2-3 levels maximum

8. Wrong Serving direction
Serving flows UPWARD. Lower serves upper. Never reverse

9. Capability vs Function confusion
Capability = what you CAN do. Function = what you DO

10. Ignoring I&M layer
Work Packages, Plateaus, Gaps have specific rules