Composable Agentic Architecture

Varun Nair

About Me

Varun Sasidharan Nair
Enterprise Architect Principal, Telstra

- Member of the Architecture Centre of Excellence (CoE) under the Chief Architect at Telstra
- Contributor to the Telstra Reference Architecture
- Architect for Telstra's Autonomous Network, with primary focus on the use of **Artificial Intelligence and Data modernisation**

Industry contribution:

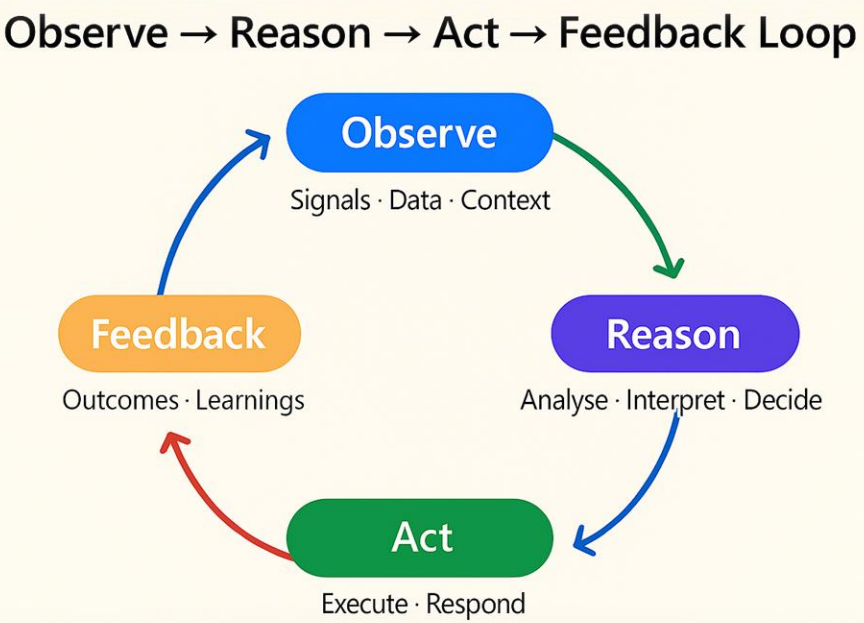
- TM-Forum (A global alliance of 800+ organizations across the connectivity ecosystem)
  - Co-Chair of Data Mission of the AI-Native Telco
  - Co-author of the Data Management Blueprint for AI-Native Telcos
  - Author and contributor to multiple TM Forum Open API standards and Blueprints

Why Good Architecture is Foundational for Agentic Evolution

Agentic Architecture

The Autonomy Leap

- AI agents are rapidly evolving, becoming impressively autonomous.
- Observe → Reason → Act → Feedback loop (cyclical)
- Agents can perceive environment and figure out “how” to achieve a goal by itself
- The possibilities are truly exciting

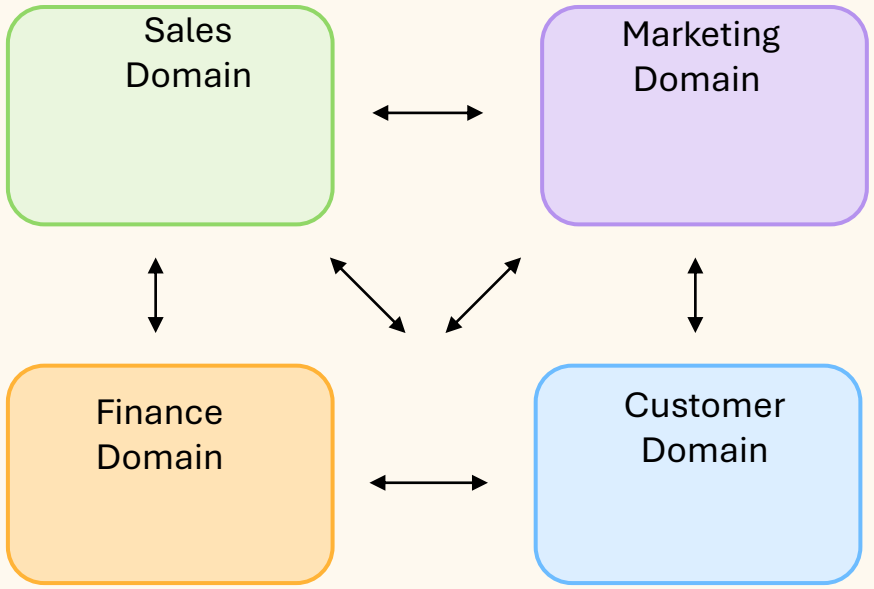


It is time to Scale!

Modular Architecture: Looking back

We worked for years to avoid "spaghetti" implementations:

- Avoided deep & uncontrolled system integrations
- Applied Domain Driven Design Principles
- Established Domains with standard interfaces and specialist softwares *E.g. sales domain, finance domain, customer domain.*
- Enabled scale without "spaghetti" implementations



Domain-Driven Design

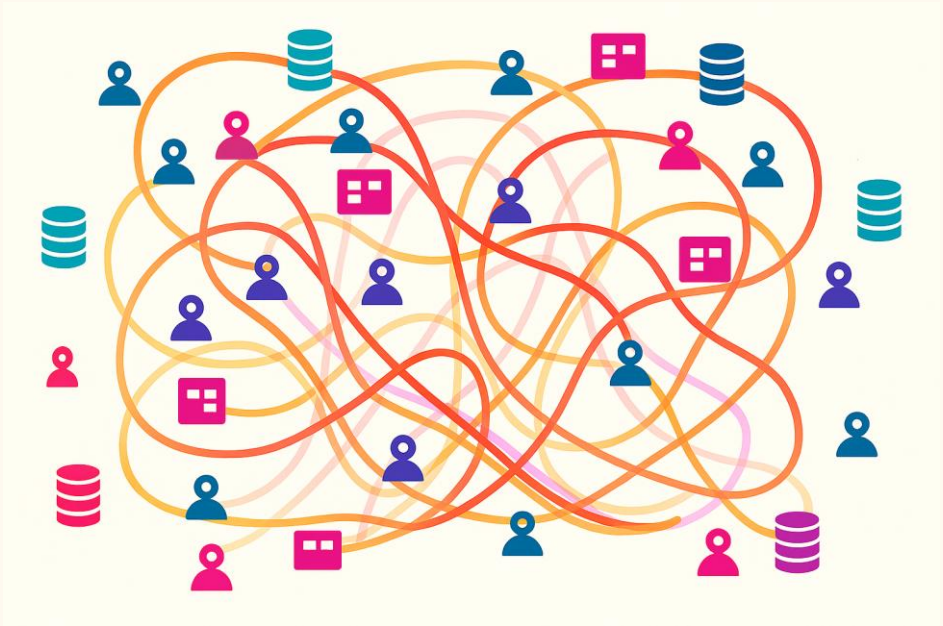
Standard Interfaces

Guard Rails

Agentic Architecture

Avoiding “Spaghetti” Implementations

- Agents with deep integrations risk return of “spaghetti” implementations
- Seamless access feels fast today
- Over a period, this become unmanageable.
- Agents may misinterpret ambiguous raw data
- Lack of guard rails risks instability.



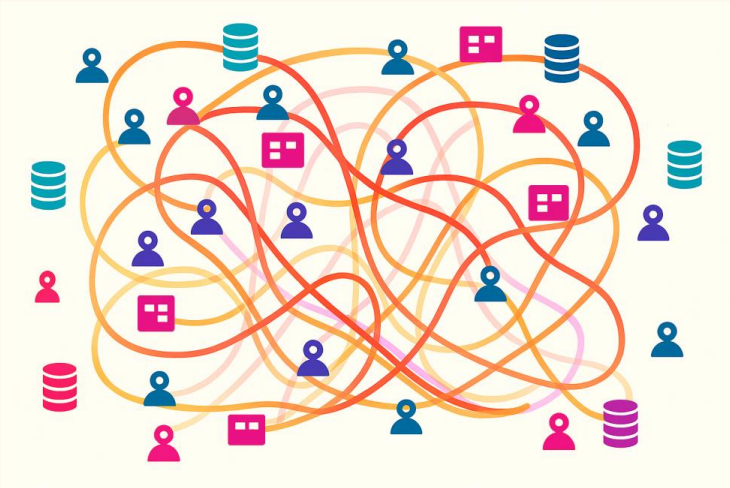
Avoid “spaghetti” implementations

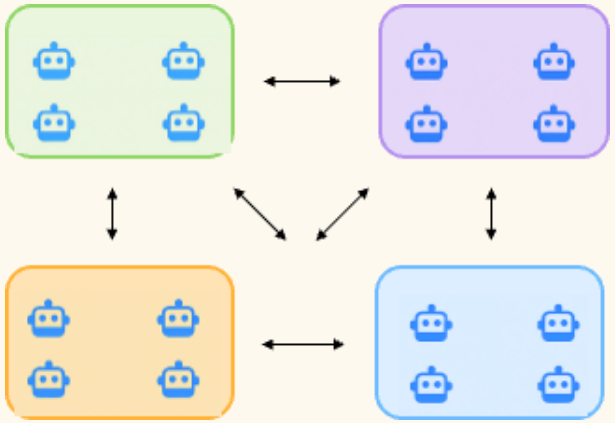
We need Architecture guardrails

Agentic Architecture

Composable Modular Agentic Architecture

Architecture Objective
Enable bold agent planning
Ensure safe agent actions
Scale agents without "Spaghetti"



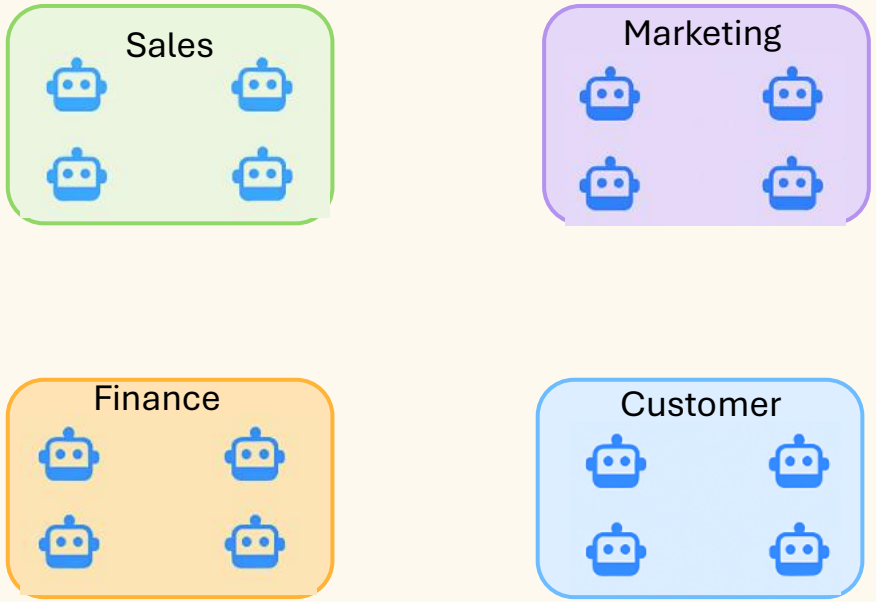


Avoid “Spaghetti” & Adopt Domain Driven Design for Agentic Architecture

Domain Driven Design (DDD)

Sovereign Domains as Strategic Anchors

- Agents to be made specialist within Domains
- Every agent now has a home and supervisor
- Grounded in the "Ubiquitous Language" of its home domain
- Specialised, governed and accountable "Digital Experts"



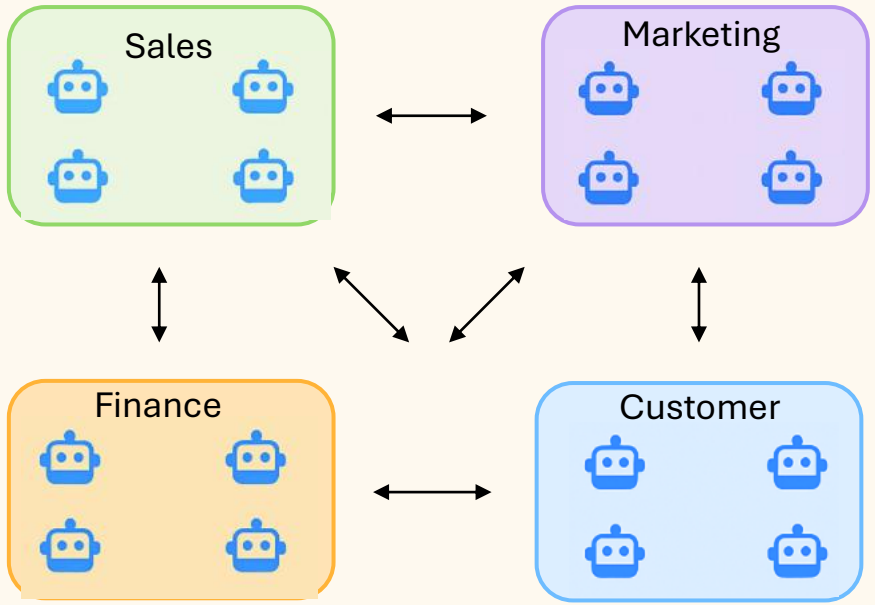
Anchor agents to each domain

Agentic Architecture

Can DDD provide agents with wider canvas for “Thinking” and “Acting”

In DDD, it is imperative that agents are given seamless cross domain collaboration.

Each Domain should:	So that Agents from other domains can:
publish standard agent ready interfaces	discover and act
share knowledge and context	think and act without ambiguity
apply guard rails	perform safer actions



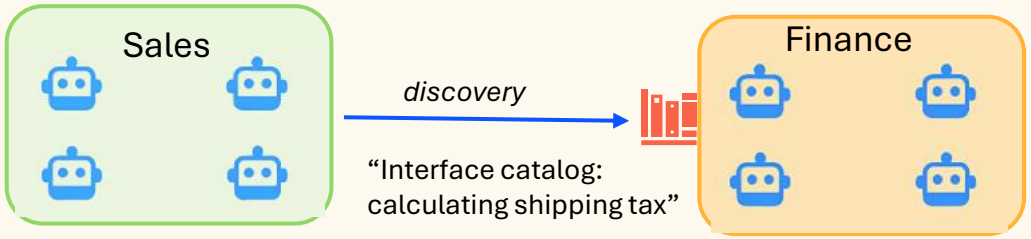
Standard Interfaces for seamless collaboration

Composable Agentic Architecture

Standardised Discovery Interface

"How can an Agent in Sales determine the tax on a shipment which is a function of the Finance Domain?"

- Every domain offer a "public catalog" of its agent-ready interfaces
- Agents in other domains discover interfaces for seamless planning
- It's transparency without the "spaghetti"



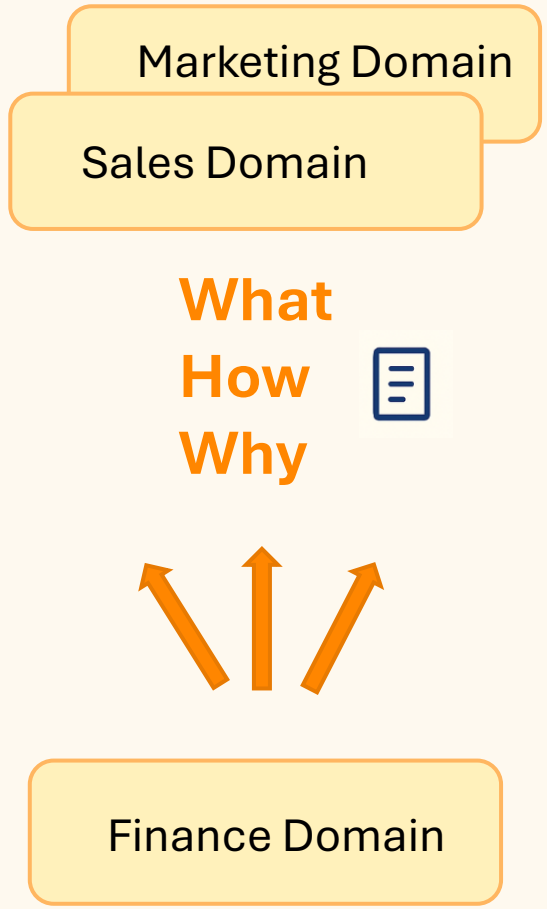
Agent ready interfaces listed in a catalog for easy discovery and planning

Agentic Architecture

Providing Context and Knowledge means Agents don't have to "Guess"

Domain adopt "Knowledge-enriched exposure" to share knowledge and context

- Each Domain provides vibrant, self-describing interfaces – WHAT, HOW, WHY
- Avoid agents from other domains duplicating logic, insights
- Avoid the need for cross domain raw data access



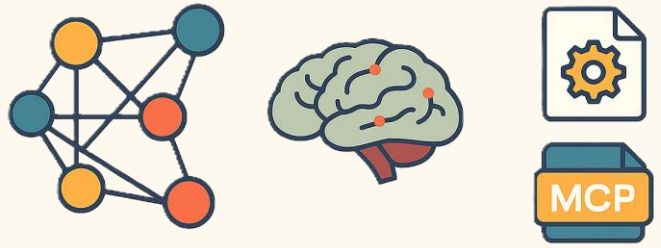
Knowledge and Context is the new currency

Agentic Architecture

Emerging Mechanisms for Knowledge-enriched exposure

There are several emerging techniques for sharing knowledge and context

- Semantic grounding by sharing context [e.g. JSON – LD]
- Analytical grounding by sharing historical context [e.g. Data Products]
- Structured representation of the knowledge of business reality [e.g. Knowledge Graphs]
- Emerging interoperability standards to share context [e.g. MCP, A2A]



Agentic Architecture

Governance and Accountability

- Governance must be "baked into" the architecture itself
- Each Domain must apply governing policies
- Continuous monitoring of agent behaviour and outcomes
- The Domain Owner is 100% accountable for agent outcome
- You can't blame the AI for the randomness of its result

Built-in AI governance

Agentic Architecture

Mechanism Design for Guard Rails

Ensure the agent's most efficient path to a goal is also its most complaint path

- Guardrails are not an afterthought, they need to be built-in
- Why Guardrails? – Agent’s most efficient path is the most complaint path
- "Shadow Environments" to run a simulation of a high-stakes move
- "Hard-Coded Invariants"– rules the agent literally cannot break

Good Architecture Is Foundational

Key takeaways:

Prevent AI Chaos through Domain Discipline

Faster, Smarter Outcomes through sharing Context and Knowledge

Clear Accountability through built-in AI Governance