Architecture Governance Artifact

Scope: Multi-product healthcare platform (Clinical, Member, Provider, Claims, Analytics)
Context: Highly regulated, EHR-centric, multiple regions, shared platform

---

1. Architecture Decision Record (ADR)

ADR-001: Multi-Product Integration & Modernization Approach

Status: Approved
Date: *TBD*
Decision Owners: Product Solution Architect, Platform Architect
Stakeholders: Product, Security, Clinical Ops, Data, SRE, Compliance

---

Context

Multiple product lines require access to clinical and member data from EHR and legacy systems. Direct integration increases coupling, compliance risk, and long-term rework. Product delivery timelines are aggressive, but architectural integrity and patient safety cannot be compromised.

Decision

Adopt a **phased strangler + façade architecture** with:

- Canonical APIs and data contracts
- Read/write separation for EHR
- Event-driven integration for cross-product data sharing
- Platform-owned shared services (identity, consent, audit, logging)

Options Considered

Option	Description	Pros	Cons
A	Direct EHR API exposure	Fast	High risk, brittle
B	Façade + strangler (chosen)	Balanced, safe, scalable	Slight delay
C	Full rewrite	Clean	Too slow, high cost

Consequences

Positive

- Controlled modernization
- Reduced compliance risk
- Enables parallel product development

Negative

- Requires governance discipline
- Initial velocity slightly reduced

Guardrails

- No direct EHR writes without approval
- All APIs must be contract-tested
- All PHI flows require ADR reference
- Shared services must be reused (no duplication)

---

2. RACI Matrix (Architecture Governance)

Activity	Product	Architect	Security	Data	Engineering	Ops	Exec
Architecture strategy	C	**A/R**	C	C	C	C	I
ADR creation & approval	C	**R**	C	C	C	C	A
API contract approval	C	**R**	C	C	**A**	I	I
Data model governance	C	R	C	**A**	C	I	I
Security design review	I	R	**A**	C	C	I	I
EHR integration approval	I	**A**	C	C	R	C	I
Shared service onboarding	C	**R**	C	C	A	C	I
Release go/no-go	C	R	C	C	**A**	R	I
Incident postmortem	I	R	C	C	A	**A**	I

A = Accountable | R = Responsible | C = Consulted | I = Informed

---

---

3. Governance Cadence & Forums

Weekly (Execution Level)

Architecture Working Group (60 min)

- API/schema changes
- Integration risks
- Upcoming ADRs
- Cross-product conflicts

Participants: Architects, Product Tech Leads, Data, Security

Biweekly (Alignment Level)

Platform & Product Sync (90 min)

- Roadmap alignment
- Shared service prioritization
- Capacity conflicts
- Dependency resolution

Participants: Product Directors, Platform Leads, Architecture

---

Monthly (Control Level)

Architecture Review Board (ARB)

- Major ADR approvals
- Security/compliance sign-off
- Technical debt review
- Resilience posture

Participants: Enterprise Arch, Security, Clinical IT, SRE, Compliance

---

Quarterly (Strategic Level)

Executive Steering Committee

- Architecture health metrics
- Risk register review
- Investment decisions
- Platform ROI

Participants: VP Product, CIO, CTO, Clinical Leadership

---

4. Architecture Health Metrics (Reported Quarterly)

Dimension	Metric
Coupling	% APIs bypassing platform (target <5%)
Resilience	MTTR, critical outage count
Security	PHI violations, audit findings
Velocity	Lead time for integration
Reuse	Shared services adoption rate
Debt	# of temporary exceptions >90 days

---

5. Operating Principles (for interviews)

1. **Disagree with data, not opinions**
2. **Govern lightly, enforce consistently**
3. **Patient safety > speed**
4. **Platforms before products (when shared)**
5. **Decisions documented or repeated**

What is an ADR?

ADR = Architecture Decision Record

An **Architecture Decision Record (ADR)** is a short, structured document that captures a **significant architectural decision**, *why it was made*, and *what consequences it has*.
It prevents re-litigation of decisions, creates alignment, and is critical in complex environments like healthcare, multi-product platforms, and regulated systems.

Why ADRs exist (in simple terms)

In large organizations (like Kaiser-style healthcare systems):

- People change roles
- Teams join later
- Decisions get questioned repeatedly
- Regulators ask *why* something was done
- Architects get blamed for undocumented trade-offs

ADRs solve this by making decisions **explicit, traceable, and auditable**.

What an ADR typically contains

A standard ADR is **1–2 pages** and includes:

1. **Title & ID**
   (e.g., ADR-001: EHR Integration Strategy)
2. **Status**
   Proposed | Approved | Deprecated | Superseded
3. **Context**
   What problem are we solving? What constraints exist?
4. **Decision**
   What we chose to do (clearly stated)
5. **Options Considered**
   Usually 2–3 realistic alternatives
6. **Consequences**
   Positive and negative impacts
7. **Guardrails / Follow-ups**
   Rules or actions that flow from the decision
8. **Owners & Date**
   Who approved it and when

Example (very short)

Context: Multiple product lines need EHR access, but direct integration creates coupling and compliance risk.
Decision: Use a façade + strangler pattern with canonical APIs.
Consequences: Slight delivery delay, but lower long-term risk and faster future onboarding.

That's an ADR.

When to create an ADR

Create an ADR when the decision is:

- Hard to reverse
- Cross-team or cross-product
- Affects security, data, or patient safety
- Likely to be questioned later
- Introduces a new pattern or exception

What ADR is NOT

❌ Not a design doc
❌ Not a wiki page
❌ Not a meeting note
❌ Not a spec

It's a **decision log**, not a design encyclopedia.

Why interviewers love ADRs

If you mention ADRs in interviews, it signals you:

- Think at **system scale**
- Manage **risk and governance**
- Can influence without authority

- Operate like a **senior / principal architect**

One-liner for interviews

“I use ADRs to document irreversible architectural decisions so teams can move fast without re-litigating trade-offs, especially in regulated healthcare environments.”
