

The Questions Behind a Good Architecture Review

Published 22 August 2026 · A practical resource from Anastasiia Solodkaia

The Questions Behind a Good Architecture Review

Architecture review is not just a technical inspection of the system. Its quality depends on which questions were raised during solution discussion.

Throughout my career I have seen companies repeating the same negative pattern: raising insufficient questions during architectural discussions. This leaves a lot of gaps in the resulting solution, leading to multiple review rounds, and potential financial loss.

This document introduces a set of questions that should be asked during every architecture review. *It can be used by different participants in the review, but mostly it is focused on business decision-makers and architects.*

It is important to clarify: typically, the company does not have the right answers to many of them before the architecture review starts. Moreover, the special knowledge required to answer them is owed by different teams, for example business, technical, compliance or security team.

So, the checklist below is intended to provide you with gentle guidance on what should be discussed during architecture review. Remember, good architecture review brings together business decisions, organisational knowledge and technical discovery.

The Questions at a Glance

If you are a business decision-maker

The questions below require a business judgement, therefore the best people to answer them are business stakeholders.

They can also expose unresolved business decisions and assumptions that would otherwise surface much later.

Look at them and check – how many of them can you answer?

- ☐ Why is this review happening now?
- ☐ Is the review proactive or reactive?
- ☐ What is the cost of delaying the change?
- ☐ What are the deadlines for this review?
- ☐ Are we prioritizing near term, long term, or both?
- ☐ Which budget is realistically available?
- ☐ What commitments have already been made?
- ☐ If we cannot optimise for everything, what takes priority?
- ☐ What is non-negotiable, and what can be traded away for now?
- ☐ What business decision should this review make possible?
- ☐ Who sponsors the review?
- ☐ Who owns the budget?
- ☐ Who makes the final decision?

- ☐ Who will execute the decision?

Questions to work through during the review

As the questions from the previous section are explored, another question may emerge: whether a full architecture review is needed at all. Sometimes the actual business need is narrower: an incident root-cause analysis, a cost comparison, a review of one specific decision, or a risk assessment

Answers to the questions below will help build the context for the review, establish its scope, and lead you towards a final decision.

- ☐ What will the review itself cost?
- ☐ What happens if we do nothing?
- ☐ Which existing architectural decisions cannot realistically be changed?
- ☐ What regulatory or security constraints apply?
- ☐ Are there parallel initiatives that could affect this review?
- ☐ What is in scope?
- ☐ What is out of scope?
- ☐ Which out-of-scope systems or initiatives still constrain the decision?
- ☐ How difficult would the decision be to reverse?
- ☐ What output does the business need?
- ☐ Who will be affected by the decision?
- ☐ Whose input is mandatory?
- ☐ Who is the final output for, and what format will be useful to them?

You can find explanations about each item of the checklist in a next section.

Why These Questions Matter

Business-led Questions

Why is this review happening now?

What triggered this review? This trigger often explains what the business is actually trying to achieve or protect.

There are large variety of the possible reasons, and the most popular are:

- a serious production incident;
- a regulatory finding or fine;
- repeated failures that have remained unresolved;
- a major upcoming launch;
- expected product or business growth;
- a need to choose right vendor or platform.

Is the review proactive or reactive?

Knowing this tells you what you need to understand before the technical work starts.

A reactive review may follow some specific negative event: a production incident, a lost client, a regulatory issue or a budget overrun.

Such review often comes with history, consequences which have already happened, questions of accountability, and sometimes some political sensitivity.

A proactive review is more likely to be driven by a future decision or an identified risk.

What is the cost of delaying the change?

Sometimes the change is necessary, but not necessarily now. This is another option typically surfacing during technical review.

What are the deadlines for this review?

The technically ideal solution is not always the right solution for the business. Under tight constraints, a faster and less elegant option may be the right short-term choice.

Is there a deadline? This deadline typically unfolds into these three important dates:

- when the review must finish;
- when the decision must be made;
- when the resulting change must be implemented.

Typically, those are not the same dates.

Are we prioritizing near term, long term, or both?

Often the right short-term choice is the one that solves today's problem and provides the door for future change to align with the long-term goal.

A decision that works for the next three months may not be the architecture you want for the next three years. But the decision that works for the next three years may take half a year to implement.

So, are we solving an immediate business problem, designing toward a longer-term solution, or targeting both and identifying clear steps of transition between the solutions?

What budget is realistically available?

There is little value in spending significant review time designing a solution the business could never fund.

The realistic budget affects which options are worth investigating.

What commitments have already been made?

Architecture solutions have to work within commitments the business has already made.

Have you made any commitments already? For example:

- a promised product or feature release;
- customer commitments;
- regulator-agreed milestones;
- market-entry dates;
- vendor contracts.

If we cannot optimise for everything, what takes priority? What is non-negotiable, and what can be traded away for now?

The business priorities navigate technical staff through the assessment of trade-offs. Architectural solutions could be optimized to fit different requirements, but most of them cannot be optimized to fit all requirements equally.

For example, these solution factors might compete with each other:

- time to market over architectural elegance;
- reliability over speed;
- lower cost over flexibility;
- reduced vendor dependency over faster implementation.

What business decision should this review make possible?

The architecture review should produce artifacts which help to make specific decisions. The report itself does not have value unless it moves the business forward.

So, what should become possible once the review is finished?

For example:

- go/no-go decision;
- build vs buy software;
- proceed with a vendor or platform;
- approve investment;
- choose between architectural options;
- decide whether a change is justified.

Who sponsors the review?

The sponsor is the person who wants the review to happen and has the authority to support it.

They open access to the people and information required.

Who owns the budget?

Who can fund the review and, more importantly, the action that may follow from it? If some potential option implementation cannot be funded, it is better to exclude it during the review.

Who makes the final decision?

Who can say the final **yes** - or veto the recommendation?

Make this explicit, otherwise a review can turn into repeated rounds of feedback and approval and eventually drag on for months.

Who will execute the decision?

The team and people who can turn decision into actions typically have to be included in the decision-making process. They might contribute significantly to discovery of potential roadblocks and limitations.

So, identify who will do it. This may be:

- an engineering team;
- an engineering lead;
- a platform team;
- a vendor;
- several teams working together.

Questions to work through during the review**What will the review itself cost?**

The review has to be worth more than it consumes. Consider more than the architect's time. The review may also consume:

- engineering time;
- business stakeholder time;
- management attention;
- time from security, compliance, operations, or vendors;
- capacity that would otherwise go to delivery.

What happens if we do nothing?

Sometimes doing nothing is genuinely cheaper. In many cases the technical decision includes this option as a part of assessment.

Which existing architectural decisions cannot realistically be changed?

Understanding which decisions cannot be changed and why helps saving on deep analysis in wrong direction. Nevertheless, the decisions already made will be the subject to review.

A great example would be the agreed cloud provider. Let's say you have already decided to use Azure. It may be the preference based on specific recommendation, or it may be a dependency-based decision, or it may be a regulatory-driven constraint.

Identify decisions that:

- genuinely cannot be changed;
- would be very expensive to reverse;
- are already committed;
- the business does not currently want to reconsider.

What regulatory or security constraints apply?

The regulatory and security constraints can eliminate otherwise attractive technical solutions right away.

Identify requirements that cannot be traded off. Depending on the business, these may include:

- regulatory obligations;
- security policies;
- certification requirements;
- data residency;
- audit requirements.

Are there parallel initiatives that could affect this review?

Technical initiatives and transformations running in parallel will probably impact each other. Independently they can produce conflicting decisions and solutions, and detecting it too late can be costly.

Look for related work already under way, such as:

- migrations;
- transformation programmes;
- vendor procurement;
- regulatory programmes;
- product strategy changes;
- organisational restructuring;
- other architecture reviews.

What is in scope?

Be explicit about the systems, products, platforms, components, or architectural decisions being reviewed.

Examples:

- user notification pipeline;
- authentication architecture;
- payment processing platform;
- customer onboarding flow.

What is out of scope?

It is better to define *known* dependencies and problems before they are found in the process of the review.

Which out-of-scope systems or initiatives still constrain the decision?

A database, shared platform, external integration, or another product might not be reviewed directly but may still constrain the architecture being assessed.

What output does the business need to make that decision?

It is necessary for the business to define the output format from the beginning, otherwise it will be hard to tell if the review is finished and has reached its goal or not.

Examples might include:

- recommendations;
- cost comparison;
- option comparison;
- risk assessment;
- prioritised actions;
- probability / impact matrix;
- executive summary;
- detailed technical findings.

How difficult would the decision be to reverse?

Ask yourself: "If this turns out to be wrong, what would we need to do to undo it?" The harder a decision is to reverse, the more scrutiny it deserves during review.

Choosing a cloud provider, a core data model, or a highly embedded platform may justify deeper analysis than selecting something that can be replaced relatively easily.

Would reversal be:

- a small configuration or implementation change;
- a moderate migration;
- another major project?

Who will be affected by the decision?

Some stakeholders may not impact or implement the decision, but they will be impacted by its consequences. It can be significant, and their input might be valuable.

Examples:

- dependent product teams;
- customer support;
- operations;
- finance;
- other technology teams.

Whose input is mandatory?

If their input is mandatory, this means they own very important requirements and constraints. It is necessary to gather them before final recommendations being produced.

They might not own the decision, but can materially change what is possible.

Depending on the situation, this may include:

- Security;
- Compliance;
- Risk;
- Legal;
- Operations;
- Data;
- Finance.

Who is the final output for, and what format will be useful to them?

The review is only useful if the people who need to act on it can understand and use the result.

An architecture review written for engineers is not the same document as one written for an executive committee. Identify the primary audience and the format that will be most useful to them.

For example:

- a short executive summary;
- an options and cost comparison;
- a risk assessment;
- a detailed technical report;

What a Good Architecture Review Should Achieve

You do not need every input detail resolved before review. Sometimes gathering that information is itself part of the review.

A good architecture review will help you to gather all business context, the knowledge distributed among different technical teams and help you to reveal the gaps in business knowledge. Its true value is in reducing uncertainty around the architectural decision you are about to make.

A good architecture review does not start with all the answers. It starts with the right questions and ends with enough clarity to make a defensible decision.