Grounded Architecture

Part II: On Being Architect

Željko Obrenović

Grounded Architecture

Željko Obrenović

This book is available at https://leanpub.com/groundedarchitecture

This version was published on 2025-06-02

Contents

1: On Being Architect: Introduction

image by azmanl from istock

IN THIS SECTION, YOU WILL: Get an overview of lessons I learned on what it means to be an architect in practice.

In this section, we begin the **second part of this book**, where theory meets practice. The first part introduced the Grounded Architecture framework, including its principles, goals, and underlying philosophy. Now, we will focus on **real-world guidance**, providing **practical tools**, **ideas**, **and inspiration** to help you apply the framework within your organization.

This part is aimed at **IT professionals** who are looking to build or improve their architecture practice, especially in complex environments where one-size-fits-all models often fall short. Instead of prescribing a rigid method, we offer a **flexible toolbox** of strategies, examples, and insights that can be adapted to your unique context. Our goal is to help you **develop a resilient and effective architectural practice** that truly fits your organization.

We will start by exploring a fundamental question: **What does it mean to be an architect in IT?** The role is **multidimensional**, requiring not only technical expertise but also strategic vision, communication skills, and leadership. Like a **Swiss Army knife**, an effective architect must be versatile, capable of shifting between coding, business discussions, and mentoring. This chapter will provide a deeper understanding of the role, highlighting what architects do, what contributes to their success, and how to grow within the profession.

Whether you are just starting your journey or you are already deep into your architectural career, the upcoming chapters are designed to offer **actionable insights and meaningful reflections**. This is not about buzzwords or stereotypes; it is about what it **actually takes** to succeed in a role that is becoming increasingly crucial every day.

1.1: Growing as an Architect

Drawing inspiration from Gregor Hohpe's book *The Software Architect Elevator*, we will explore how architects grow and evolve. Hohpe introduces a useful metaphor: the **three-legged stool of architect development**, which consists of:

- **Skills** – the ability to understand and work with both technology and people,
- **Impact** – making a measurable difference in the business,
- **Leadership** – guiding others, setting direction, and raising standards.

For an architect to thrive, all three legs must be strong. If one is too short, the stool tips over:

- If you have **skills and impact** but lack leadership, you may hit a **glass ceiling**, unable to increase your influence.
- If you have **leadership without real impact**, you risk becoming an **ivory tower architect**, disconnected from what truly matters.
- If you have **impact and leadership** but lack skills, your decisions may be **impractical or uninformed**, missing critical technical depth.

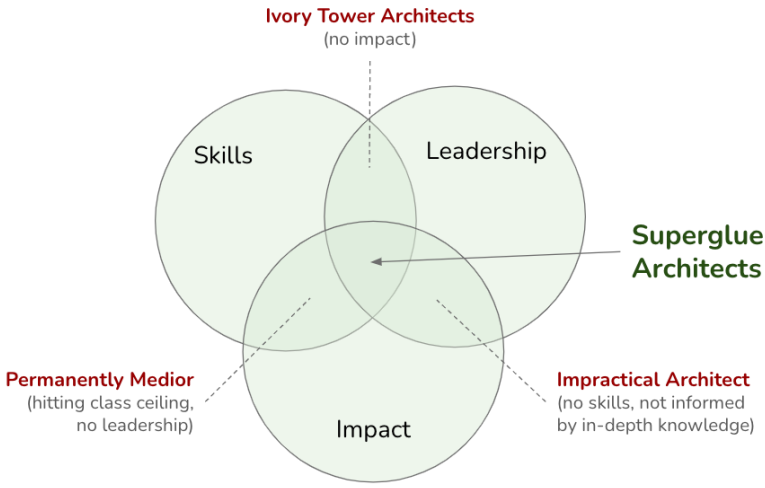


***Figure 1**: Architects must have a minimal "length" of all "legs" to be successful.*

1.1.1: Skills: Building a Strong Foundation

At its core, architecture relies on **expertise**—both technical and interpersonal. A successful architect combines in-depth knowledge with strong abilities to **communicate**, **collaborate**, **and influence**.

A well-rounded skill set typically includes:

- **Technical depth**: proficiency in modern cloud platforms as well as legacy systems,
- **Communication skills**: the ability to reassure nervous developers and translate business language into actionable technology plans,
- **Product understanding**: insight into how systems provide value,
- **Business insight**: an understanding of priorities and trade-offs, often represented in spreadsheets,

- **Decision-making**: the capability to navigate complexity confidently, even when there isn't a clear answer.

For a more detailed exploration, please refer to the Skills section.

1.1.2: Impact: Delivering Real Value

Architects play a crucial role in driving meaningful change. In a business context, this means that your work should create measurable value, whether through cost savings, faster delivery, improved systems, or smarter strategies.

Examples of impactful architecture include:

- Bridging the gap between business goals and technical execution.
- Streamlining processes to boost team productivity.
- Making systems more efficient through smart cost optimizations.
- Developing technology strategies that align with long-term goals.
- Supporting product delivery, enabling teams to move faster with fewer obstacles.
- Fostering innovation, not only in ideas but also in execution.

For practical examples, please see the Impact section.

1.1.3: Leadership: Raising the Bar

Architects do more than just design systems—they lead people. Senior architects not only influence the technology stack but also help shape the culture, impacting how teams think and work.

Key leadership traits include:

- Acting as a **role model**, setting high standards for both technology and teamwork.

- Leading **high-stakes initiatives** that address core business challenges.
- **Contributing to the tech community** through writing, speaking, or open-source projects.
- Fostering a culture of **continuous learning and improvement**.
- **Mentoring** future architects, aiding them in avoiding common pitfalls and accelerating their growth.

For experienced architects, leadership isn't optional—it's essential. For more insights, see Leadership Traits.

1.2: Thinking Like an Architect

In large organizations, architects act as the **connective tissue**—bridging strategy and execution, business and technology, as well as teams and outcomes. This role is often referred to as "**organizational superglue**" because architects unify the various components of a complex system.

However, being the glue doesn't mean being invisible. It involves being intentional—**balancing vision with pragmatism** and ambition with realism. Architects must navigate uncertainty, ask difficult questions, and bring together diverse perspectives. Additionally, they need to master the delicate balance of:

- **Curiosity**: remaining open to new ideas,
- **Doubt**: questioning assumptions,
- **Vision**: striving towards a desired future,
- **Skepticism**: ensuring that ideas are grounded in reality.

This mindset enables architects to lead **sustainable change**, rather than merely pursuing exciting innovation. For more insights, see Superglue and Balancing Forces.

1.3: Career Growth for Architects

Most successful architects start their careers as engineers. This hands-on experience provides the **technical foundation** necessary for making high-level decisions later on. As architects advance in their careers, they extend their influence—initially across teams, then departments, and eventually throughout the entire organization.

Over time, architects may transition into roles such as:

- **Principal Architect** – leading complex initiatives,
- **Engineering Director** – managing teams and setting the technical direction,
- **CTO** (Chief Technology Officer) – aligning technology with the overall business strategy at the highest level.

Career growth involves more than just acquiring knowledge—it also means **broadening your perspective**, deepening professional relationships, and learning to communicate effectively in both technical and business contexts.

For more information on how architects evolve in their careers, see Architects' Career Paths.

2: Thinking Like an Architect: Architects as Superglue

image by pagadesign from istock

IN THIS SECTION, YOU WILL: Understand the view on architects as superglue (people who hold architecture, technical details, business needs, and people together across a large organization or complex projects) and get valuable tips on developing "superglue" abilities.

KEY POINTS:

- Architects in IT organizations should develop as "superglue," people who hold architecture, technical details, business needs, and people together across a large organization or complex projects.
- Architects need to be technically strong. But their unique strengths should stem from being able to relate technical issues with business and broader issues.

To succeed as an IT architect, it's not enough to simply master tools, frameworks, or technologies. Achieving success requires developing the **right mindset**—a blend of beliefs, attitudes, and mental models that influence how you perceive challenges and approach your work.

2.1: The Superglue Mindset: What It Takes to Be an Effective IT Architect

One of the most powerful metaphors for cultivating this mindset is that of the "**superglue" architect**. This concept captures the essence of what great architects do, particularly in complex, dynamic organizations where collaboration and alignment can be more challenging than coding itself.

2.1.1: What Does It Mean to Be "Superglue"?

The **superglue metaphor** emphasizes the role of architects as the **binding force** connecting various components of an organization—people, teams, systems, strategies, and goals. This idea was popularized by Adam Bar-Niv and Amir Shenhav from Intel[1], who observed that, while not everyone can be a superhero in a crisis, organizations greatly benefit from individuals who quietly and consistently hold everything together.

Tanya Reilly[2] builds on this concept in her discussions about "glue work"—the invisible but vital contributions that ensure collaboration, coherence, and continuity. Similarly, Gregor Hohpe[3] describes modern architects as those who navigate the "**Architect Elevator**," seamlessly moving between the **penthouse** (where business strategy is defined) and the **engine room** (where technical implementation occurs). Superglue architects thrive in this in-between space, adding value by ensuring alignment across different organizational levels.

2.1.2: The Role of a Superglue Architect

Superglue architects take on more than just writing technical specifications or selecting the right cloud solutions. They play a **connective role** across four key areas:

[1] https://saturn2016.sched.com/event/63m9/cant-find-superheroes-to-help-you-out-of-a-crisis-how-about-some-architecture-and-lots-of-superglue

[2] https://noidea.dog/glue

[3] https://architectelevator.com/

1. **Architecture** – Defining technical direction, ensuring consistency, and promoting scalable design choices.
2. **Technology** – Having a deep understanding of tools, systems, and platforms so that they can engage engineers meaningfully.
3. **Business Needs** – Translating between strategic goals and technical feasibility, ensuring that solutions align with the organization's actual requirements.
4. **People and Teams** – Facilitating communication, resolving conflicts, mentoring team members, and ensuring that everyone is working towards the same objectives.

In essence, these architects serve as the **organizational connective tissue**. Their value lies not only in their problem-solving abilities but also in their capacity to **link problems to the right solutions**—and to the right people.

2.1.3: The Superglue Mindset in Action

Cultivating a superglue mindset involves adopting a set of perspectives and habits that go beyond technical skills. It includes:

- **Curiosity**: Always seeking to understand how things work, from legacy systems to new business initiatives.
- **Empathy**: Understanding the pressures and concerns of developers, product managers, and executives alike.
- **Adaptability**: Thriving in ambiguity and helping to clarify direction when others are still forming opinions.
- **Strategic Thinking**: Seeing the big picture and helping others connect their work to broader objectives.
- **Humility**: Recognizing that your most significant contributions might not be a brilliant design, but rather a well-facilitated conversation.

2.1.4: Why This Matters

In complex organizations, architecture is not only about systems; it's also about **relationships**, **priorities**, **and timing**. An architect with a "superglue" mindset acts as a **force multiplier**, enhancing the efforts of teams, bridging gaps between silos, and fostering cohesion in environments that might otherwise fragment under pressure.

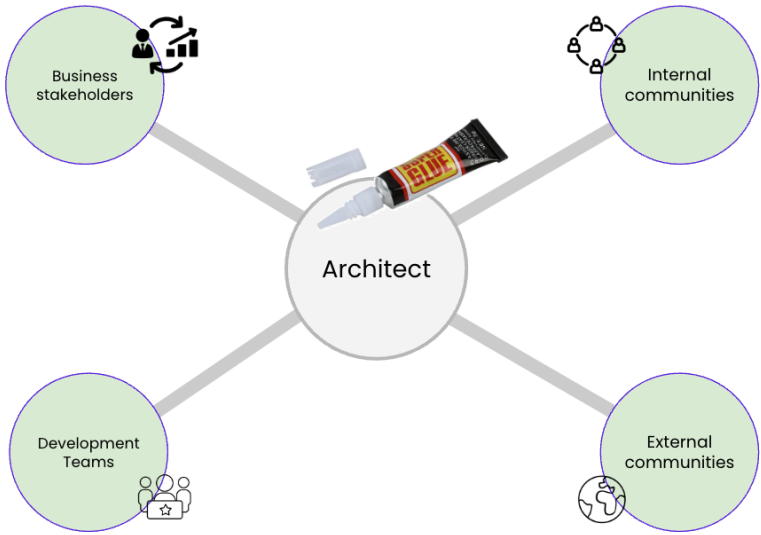
This mindset is especially crucial when managing change. Whether adopting a new platform, rethinking system architecture, or aligning business and IT strategies, the work of change takes place in the **gaps**—and that's where superglue architects excel.

If you're building your architecture practice or mentoring future architects, instilling the **superglue mindset** is one of the most effective ways to ensure long-term, sustainable success—not just for individuals, but for the entire organization.

2.2: Superglue Architects: Keeping the Organization United

In IT organizations, deep technical knowledge is essential, but it is not what truly distinguishes great architects. What sets **superglue architects** apart from other technical specialists is their ability to **connect people**, **priorities**, **and systems** through exceptional **relational skills**.

Superglue architects are more than just problem solvers—they are **connectors**. They communicate clearly, negotiate thoughtfully, and influence strategically. These soft skills, combined with their technical credibility, make them **invaluable for ensuring coherence**, **alignment**, **and momentum** throughout the organization.



Architects as Superglue

Figure 1: *Architects serve as superglue, connecting development teams with business stakeholders and linking their teams with internal and external communities.*

Figure 1 illustrates the **superglue metaphor**: architects act as the **binding agent** between the often-disconnected parts of a modern organization. Here's how that role plays out in practice:

2.2.1: Developer Whisperers

Superglue architects maintain a **close relationship with development teams**, deeply understanding their needs, roadblocks, and priorities. They don't just impose architectural decisions—they collaborate, explain, and ensure that solutions are not only sound but also **practical and empowering** for developers.

2.2.2: Tech-to-Business Translators

Architects must be fluent in both technical and business languages. They help stakeholders understand how a technical decision will impact timelines, revenue, or customer experience. In doing so, they bridge the gap between **technology potential and business value**.

2.2.3: Cross-Functional Diplomats

Superglue architects work across various departments—operations, finance, marketing, HR—not just within IT. They ensure that technical strategies align with broader organizational goals and constraints. This cross-functional engagement enables them to champion **solutions that are viable, sustainable, and strategically sound**.

2.2.4: Community Connectors

Internally, architects are **active contributors to knowledge-sharing communities**, facilitating learning, promoting best practices, and staying attuned to emerging challenges. They help **enhance the capabilities of the entire organization** by engaging with others and fostering collaboration.

2.2.5: Industry Influencers

Outstanding architects also look outward. By participating in external forums, publishing articles, speaking at conferences, or contributing to open-source projects, they bring **outside perspectives** to their organizations. This external visibility positions the organization as a **leader in the industry**.

2.2.6: Why Superglue Architects Matter

While superheroes might swoop in for a dramatic rescue, **superglue architects quietly prevent crises from occurring in the first place**. They anticipate friction points, foster alignment, and ensure that systems, strategies, and people are working in unison.

Their presence ensures:

- **Stability** across fast-moving teams and initiatives,
- **Clarity** amidst conflicting priorities,
- **Continuity** in a rapidly evolving technical landscape.

Think of them as the **organizational equivalent of duct tape**—invisible when effective, indispensable during challenges, and always ready to hold things together when complexity threatens to unravel them.

Superglue architects don't just connect systems—they **connect vision with execution**, people with purpose, and complexity with clarity. This mindset elevates them from good architects to great ones.

2.3: Superglueing in Action #1: Aligning Conflicting Organizational Goals

In many IT organizations, tensions often arise between key areas: **technology**, **product**, **organizational structure**, **and business strategy**. These areas tend to evolve at different speeds and often have conflicting priorities. Without proper alignment, the result can be poor decisions, increased complexity, stalled initiatives, and missed opportunities.

This is where **superglue architects** come in. They don't create more bureaucracy or control; instead, they **foster alignment and build bridges** across functions. Their goal is to bring these areas closer together, creating a more cohesive, responsive, and adaptive organization.

Architect alignment

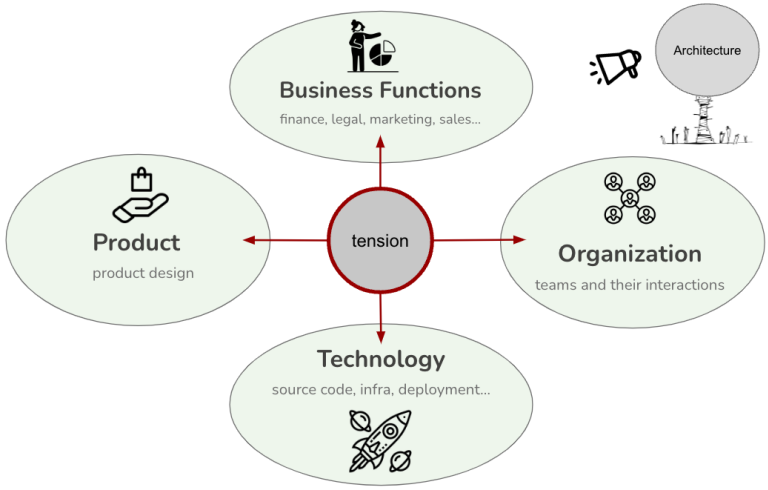
image by flamingoimages from istock

The core value of superglue architects lies in their ability to **align**

product, **business**, **technology**, **and organizational goals**, helping each function move in sync like a well-coordinated team rather than a fragmented crowd.

2.3.1: Understanding the Tensions

While it's ideal for business, technology, product, and organizational functions to evolve together, the reality is often more chaotic. These domains can act like separate dancers moving to different music. When misaligned, even small efforts can become tangled, leading to delays, inefficiencies, and friction.



Tensions diagram

***Figure 2**: Common misalignments across business, product, technology, and organizational domains.*

Let's examine a few real-world examples of these tensions:

2.3.1.1: Technology vs. Organization

- **Scenario**: A company employs a monolithic architecture to centralize operations, aiming for control and consistency. However, the development teams struggle to move quickly, as every change requires coordination across multiple groups.
- **Impact**: Innovation stalls, bottlenecks grow, and autonomy disappears. A single bug fix can trigger delays across unrelated teams.

2.3.1.2: Product vs. Technology

- **Scenario**: The product teams focus on user experience across systems, but engineering has adopted microservices organized by internal domains. A seemingly simple UI feature may require coordination across several backend services.
- **Impact**: What should be a minor enhancement turns into a game of "Whac-A-Mole," slowing delivery and frustrating teams.

2.3.1.3: Business vs. Product

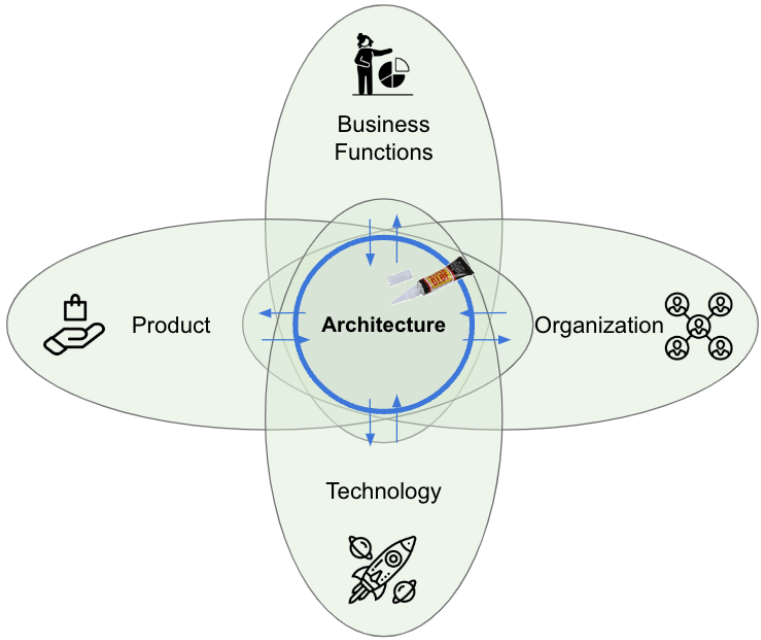
- **Scenario**: The business alternates between cost-saving initiatives and pushes for innovation. One month, the focus is on reducing expenses; the next, it's on aggressive feature delivery and platform overhauls.
- **Impact**: Product teams struggle to plan, resources shift unexpectedly, and confusion and burnout increase.

2.3.1.4: Organization vs. Business

- **Scenario**: Business goals necessitate seamless collaboration—such as launching a product—but the organization is siloed into departments like IT, marketing, and finance, each with isolated goals.
- **Impact**: Strategic initiatives stall, teams miscommunicate, and projects slip due to handoffs and misaligned incentives.

2.3.2: How Superglue Architects Reduce Tension

Superglue architects are uniquely positioned to **identify**, **mediate**, **and resolve these tensions**. They understand the entire system—including its people, processes, and priorities—and help align them into a cohesive whole.



Reducing tension diagram

***Figure 3**: Architects sit at the intersection of business, product, technology, and organizational functions to reduce misalignment.*

Here's how they achieve this:

2.3.2.1: 1. Aligning Technology with Business Goals

- **Scenario**: Business leaders want to cut costs quickly by moving to the cloud, but the tech team knows that rushing could create technical debt.
- **Architect's Role**: Facilitate a migration strategy that balances speed and sustainability. Translate technical risks into business terms, helping executives make informed, long-term decisions.

2.3.2.2: 2. Balancing Product Ambition with Organizational Capacity

- **Scenario**: Product teams want to launch features based on customer feedback, but delivery teams are overloaded.
- **Architect's Role**: Bring stakeholders together to adjust priorities or restructure teams. Suggest DevOps practices or platform changes that can support faster delivery.

2.3.2.3: 3. Translating Technical Jargon for Business Leaders

- **Scenario**: The business wants to reduce infrastructure costs, but engineers are planning major upgrades for security and scalability.
- **Architect's Role**: Explain why short-term savings could lead to long-term risks. Reframe technical needs as business continuity concerns to ensure critical work stays funded.

2.3.2.4: 4. Coordinating Product and Technology Evolution

- **Scenario**: Engineering wants to transition to a new architecture, while Product is still working with the old one.
- **Architect's Role**: Propose a clear transition strategy that allows for gradual evolution, ensuring that both teams are aligned on goals and timelines.

2.4: Supergluing in Action #2: Aligning Discussions Around Problems, Solutions, and Implementation

In complex technical environments, disagreements are inevitable. However, many of these conflicts arise not because people can't find common ground, but because they are **talking past each other**.

One powerful mental model for resolving these conflicts is derived from the Theory of Constraints (TOC)[4], developed by Eliyahu M. Goldratt. TOC is a management philosophy focused on improving performance by identifying and addressing the most critical constraint in a system.

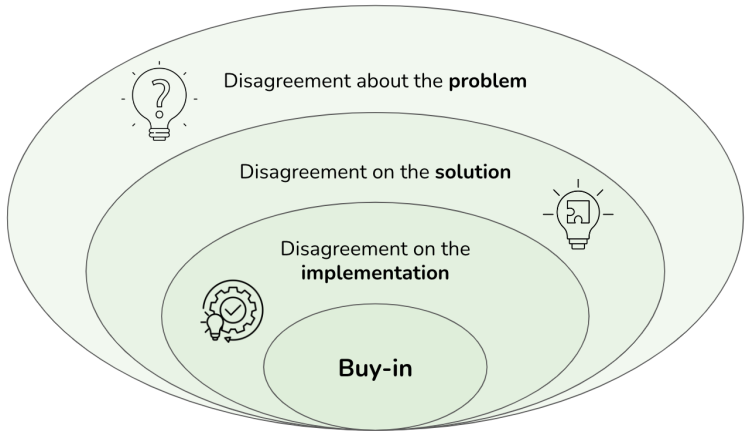
TOC image

image by peopleimages from istock

[4] https://en.wikipedia.org/wiki/Thinking_processes_%28theory_of_constraints%29

One important aspect of Goldratt's thinking that is particularly beneficial for **superglue architects** is his model for **resistance to change**. It identifies three key areas where disagreement typically arises:

1. What is the **problem**?
2. What is the **solution**?
3. How should we **implement** it?



Disagreements diagram

***Figure 4**: Goldratt's model of resistance: disagreement about the problem, the solution, and the implementation.*

Misalignment at any of these levels can derail collaboration. The **real danger** occurs when people are unaware of *which level* they are actually disagreeing about. Superglue architects can play a pivotal role in diagnosing and resolving these mismatches.

2.4.1: 1. Disagreement About the Problem

"What should we change?"

Different stakeholders often define the same issue in various ways, especially in large organizations. This divergence leads teams to waste time fixing **symptoms** rather than addressing root causes.

Example: An e-commerce platform is experiencing slow response times.

- The **database team** blames inefficient queries.
- The **network team** blames latency.
- The **backend team** suspects code issues.
- The **UX team** thinks the slowness is caused by frontend problems.

Each group starts optimizing their own area without a shared understanding of what the core problem is. The result? Disconnected fixes, wasted efforts, and ongoing performance issues.

2.4.2: 2. Disagreement About the Solution

"What should we change to?"

Even when teams agree on the problem, they may have differing opinions on the best way to resolve it due to varying experiences, biases, or risk tolerances.

Example: After agreeing that the database is the bottleneck:

- DBAs want a **server upgrade**.
- Developers suggest **query optimization**.
- Frontend developers see this as an opportunity to **rebuild the UI layer**.
- Operations propose **horizontal scaling**.

These competing solutions often vie for attention (and budget), delaying progress and sometimes introducing **new issues**.

2.4.3: 3. Disagreement About the Implementation

"How do we make the change happen?"

Even with consensus on the problem and solution, the "how" can be just as contentious—especially across departments with different timelines, KPIs, or constraints.

Example: Everyone agrees to optimize database queries.

- Developers want to **rewrite everything in one sprint**.
- Operations prefers a **phased rollout**.
- QA raises concerns about **test coverage**.
- Leadership wants to see results within **two weeks**.

If unresolved, these differences can **stall delivery**, introduce risk, or create friction that undermines the initiative.

2.4.4: The Most Dangerous Misalignment: Talking at the Wrong Level

Often, people believe they're disagreeing about a solution, but they may not even be aligned on the problem.

This miscommunication leads to endless debates, with teams discussing different issues under the same heading.

2.4.4.1: Example 1: Mistaking a Problem for an Implementation Debate

A development team argues over how to fix slow performance:

- One side wants code refactoring.
- Another wants new servers.
- A third suggests frontend optimization.

However, no one agrees on *what's actually causing* the slowness. They're **debating implementation without a shared understanding of the root issue**—similar to arguing over medication before diagnosing the disease.

2.4.4.2: Example 2: Debating Solutions Without Clarifying the Problem

A security team is divided:

- One group wants a new firewall.
- Another prefers better encryption.

The conflict appears to be solution-based. However, one group is focused on **external threats**, while the other is concerned about **internal breaches**. They aren't aligned on **what risk they're trying to mitigate**, so the solution conversation goes nowhere.

2.4.5: The Role of a Superglue Architect

Superglue architects play a vital role in helping teams navigate discussions, ensuring clarity and alignment. Here's how they do it:

2.4.5.1: 1. Clarify the Problem First

Architects begin by ensuring that the team explicitly states and agrees on the problem at hand. Tools such as root cause analysis, problem statements, or system maps can be useful in this process.

2.4.5.2: 2. Separate Problem, Solution, and Implementation

Architects assist teams by identifying the level of discussion they are engaged in. Are they discussing what's wrong, what should be done, or how to implement it? Recognizing this distinction often helps to reduce conflict.

2.4.5.3: 3. Facilitate Neutral Communication

To foster understanding, architects utilize visuals, models, and structured dialogue. They also help translate jargon or technical nuances across different team members.

2.4.5.4: 4. Introduce Multiple Perspectives

As Gregor Hohpe points out, effective architects help others understand that what may seem like disagreement is often just **different perspectives on the same system**. Sometimes, introducing a new perspective or model can transform conflict into valuable insight.

By keeping conversations focused and grounded, architects prevent teams from becoming stagnant. They promote alignment, highlight the real issues, and ensure that the team's energy is directed toward **addressing the right problems in the right way**.

2.5: Superglueing in Action #3: Navigating Organizational Conflicts

In complex organizations, **conflict is not a sign of failure—it's a sign of interdependence**. When multiple teams, priorities, and personalities come together, disagreements are inevitable. What truly matters is how these conflicts are **understood, navigated, and resolved**.

IT architects, positioned at the intersection of business and technology, often find themselves in the midst of such tensions. However, rather than acting as referees, great architects serve as **facilitators—guiding teams toward clarity, collaboration, and shared purpose**.

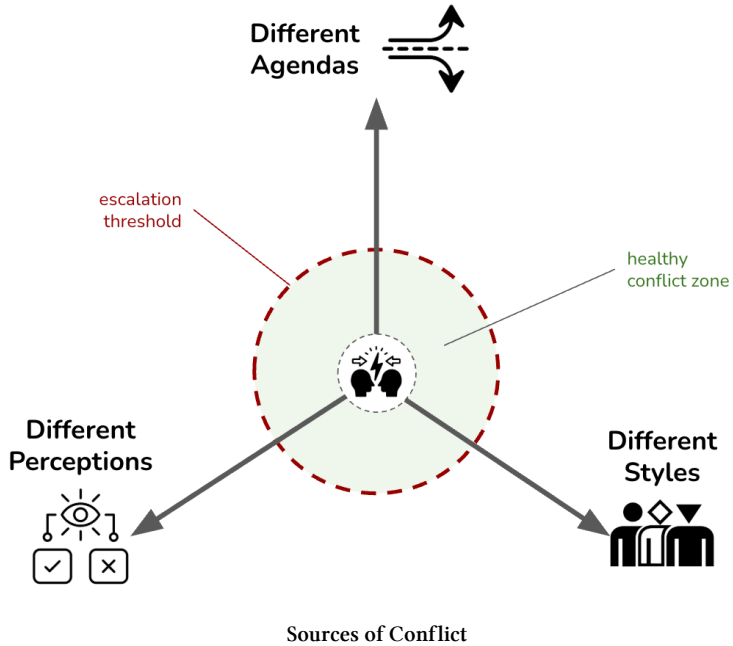


Figure 5: *Common sources of workplace conflict include different agendas, diverse perceptions, and contrasting personal styles.*

According to conflict management expert **Jeff Weiss**, workplace conflict often arises from three key sources:

1. **Conflicting agendas**
2. **Divergent perceptions**
3. **Contrasting personal styles**

Understanding these root causes enables architects to reduce friction, build trust, and maintain alignment across diverse groups.

2.5.1: 1. Managing Conflicts from Conflicting Agendas

Every team has its own set of priorities—and they are often all valid:

- **Product** wants new features to stay competitive.
- **Engineering** aims to reduce technical debt for long-term stability.
- **Operations** focuses on reliability and cost-efficiency.

These goals aren't inherently in conflict, but without coordination, they can create a tug-of-war.

Architect's Role: Architects help **reframe the discussion from "either/or" to "yes/and."** For example, they might propose a **phased roadmap** that delivers incremental features while also addressing critical technical debt. By focusing conversations on **shared business outcomes** rather than territorial disputes, architects uncover compromises that advance multiple priorities simultaneously.

2.5.2: 2. Addressing Conflicts from Divergent Perceptions

Misalignment often arises not from opposing goals, but from **different interpretations** of the same conversation.

Example: A business leader leaves a meeting thinking the team committed to delivering a product in three months, while the engineers believe they only agreed to an exploratory phase.

Architect's Role: Architects act as **alignment facilitators**, ensuring that expectations are clearly articulated and agreed upon. They might:

- Create **shared documentation** (e.g., Architecture Decision Records, meeting summaries)
- Use **visual artifacts** like roadmaps or architecture diagrams
- Confirm understanding by **summarizing key decisions** at the end of meetings

They also encourage teams to **surface and test assumptions early**, preventing costly misinterpretations later on. Tools like architecture analytics can help reveal different views and help unify them through data.

2.5.3: 3. Navigating Conflicts from Contrasting Personal Styles

Not all conflict is about content; sometimes it's about **how** people prefer to work.

- One person thrives in a **structured**, **deadline-driven** environment.
- Another prefers **flexibility and emergent decision-making**.
- A strategist may speak in broad visions, while an engineer focuses on details.

These mismatches can create friction even when the goals are aligned.

Architect's Role: Architects often serve as **translators between styles**. They might:

- Translate **high-level strategy** into **concrete technical plans**

- Facilitate communication between **big-picture thinkers** and **detail-oriented implementers**
- Moderate meetings to ensure all voices are heard, including those with differing communication preferences

Tools like the culture map can help architects understand and adapt to individual and team working styles.

2.6: Superglue Architects as Catalysts for Constructive Conflict

Architects don't need to act as conflict mediators, but they are uniquely positioned to:

- **Recognize when tensions are emerging**
- **Diagnose their underlying causes**
- **Create space for resolution that benefits the broader organization**

By applying structured decision-making, clarifying communication, and honoring diverse work styles, superglue architects transform conflict from a source of friction into a **catalyst for alignment and progress**.

Their ability to work across agendas, perceptions, and personalities helps teams **navigate disagreement without losing direction**—and builds trust in the architect's quiet but powerful role as the organization's connective tissue.## Superglueing in Action #3: Navigating Organizational Conflicts

In complex organizations, **conflict is not a sign of failure—it's a sign of interdependence**. When multiple teams, priorities, and personalities converge, disagreements are inevitable. What matters is how these conflicts are **understood, navigated, and resolved**.

IT architects, sitting at the intersection of business and technology, often find themselves in the middle of such tensions. But rather than acting as referees, great architects serve as **facilitators—guiding teams toward clarity, collaboration, and shared purpose**.

2.7: Final Thoughts on Superglue Impact: Keeping Everyone on the Same Page in a Stormy Sea

In fast-paced and turbulent organizations, **misalignment is not just a communication issue—it's a business risk**. Architects, especially those who adopt a "superglue" mindset, play a critical role in guiding organizations through complexity. They help teams stay **in sync**, ensure that **technology is aligned with real business needs**, and maintain **strategic cohesion among diverse stakeholders**.

Architects must remain closely connected to technology—not for its own sake, but to ensure it serves the **business**, **customers**, **and product vision**. They act as interpreters, facilitators, and connectors, **keeping everyone on the same page** and rowing in the same direction, even during organizational storms.

Stormy Sea

image by mbbirdy from istock

When alignment breaks down, serious risks emerge. Here are some pitfalls that superglue architects are uniquely positioned to **prevent or correct**:

2.7.1: Building the Wrong Products

When technical teams operate on **false assumptions**, they may end up creating products that nobody needs—like making snow boots for the Sahara. Architects ensure that teams stay focused on **real user needs and business priorities**, rather than getting lost in internal enthusiasm.

2.7.2: Misaligned Prioritization

Without shared metrics and clear direction, teams may waste time on interesting but low-value projects. Architects help by linking initiatives to **measurable outcomes**, allowing teams to prioritize efforts that genuinely drive results.

2.7.3: Unexpected Delivery Delays

Misunderstood complexities and hidden dependencies can cause projects to stall. Architects help identify **unknowns early**, align stakeholders on realistic timelines, and minimize costly surprises.

2.7.4: Duplication of Effort

In uncoordinated environments, teams often end up replicating the same functionality across different silos. Architects mitigate this by fostering **shared architectural awareness and system reuse**, which increases efficiency and reduces waste.

2.7.5: Unnecessary Complexity

Sometimes systems are overengineered to accommodate every possible use case—like using a Swiss Army knife when a simple spoon would suffice. Architects guide teams to **focus on what is essential**, creating solutions that are appropriately sized and sustainable.

2.7.6: Overengineering

Without proper oversight, teams may develop technically impressive yet overly elaborate systems. Imagine using a monster truck for grocery shopping—flashy, but excessive. Architects advocate for **simplicity that maintains purpose**.

2.7.7: Underengineering for Reality

Conversely, underestimating complexity can lead to fragile and unscalable systems. Architects recognize when **essential complexity must be embraced** and guide teams to build resilience without unnecessary complications.

2.7.8: Low-Quality Products

When complexity is mishandled or expertise is lacking, the result is brittle and unreliable systems. Like a dollar-store umbrella in a hurricane, these solutions fail under pressure. Architects emphasize quality by **promoting simplicity**, **clarity**, **and technical robustness**.

2.7.9: Complicated Dependencies Between Teams

Poor design of teams and systems can lead to tangled coordination and bureaucratic bottlenecks. Architects assist in creating **modular architectures and organizational structures** that promote autonomy and speed.

2.7.10: Fragile, Unsustainable Team Structures

Relying on a few specialists for critical systems poses significant risks. If key personnel leave, progress can grind to a halt. Architects encourage **resilience through redundancy**, mentorship, and strategic team design.

2.7.11: The Real Impact of Superglue Architects

Architects don't need to be vocal to make a significant impact. Their effectiveness is often reflected in what *doesn't* happen:

- No more "surprise" delays
- Fewer reworks or redundant systems
- Less friction between departments
- Greater confidence in delivery

They are the quiet force that keeps strategy, technology, and execution **interconnected through strong communication and shared purpose**. Amid shifting priorities and organizational change, **they keep the boat afloat and everyone heading in the same direction**.

2.8: To Probe Further

- Thinking Like an Architect[5], by Gregor Hohpe, 2024
- Architects See More Dimensions[6], by Gregor Hohpe, 2020
- Architects Zoom In and Out, See Different Things[7], by Gregor Hohpe, 2020
- Architects Look For Causality[8], by Gregor Hohpe, 2020
- Architects See Shades of Gray, Look for Balance[9], by Gregor Hohpe, 2020
- Here's why enterprise IT is so complex[10], by Gregor Hohpe, 2018

[5] https://www.infoq.com/articles/thinking-like-architect/
[6] https://architectelevator.com/architecture/multiple-dimensions/
[7] https://architectelevator.com/architecture/architects-zoom/
[8] https://architectelevator.com/architecture/architects-causality/
[9] https://architectelevator.com/architecture/architects-shades-of-gray/
[10] https://architectelevator.com/architecture/it-complexity/

2.9: Questions to Consider

Being a superglue architect means constantly developing and redefining your role to benefit a changing organization. Ask yourself the following questions:

- *How well do you think you currently embody the characteristics of a "superglue" architect? Which areas could you improve on to become more effective in this role?*
- *Reflect on your ability to connect the "business wheelhouse" and the "engine room" within your organization. How effectively do you bridge the gap between technical issues and business needs?*
- *How strong are your relationships with developer teams, local business stakeholders, and broader internal communities? How could you strengthen these connections?*
- *How much external visibility do you currently have? How could this be enhanced to promote the flow of ideas into and out of the organization?*
- *Can you identify specific instances of tension between your organization's technology, product, organization, and business functions? What caused this tension, and how was it addressed?*
- *How could your current architecture aid in reducing tension between these functions?*
- *Have you witnessed the architecture sitting on the side, being ignored? If so, what steps can you take to actively involve architecture in decision-making processes?*
- *Are conversations between the technical, product, organizational, and business functions encouraged and facilitated within your organization? If not, how might they be initiated and supported?*
- *Considering the three legs of a successful architect (skills, impact, leadership), which are your strongest? Which might need more development?*

3: Navigating Career Paths in Architecture

image by richvintage from istock

IN THIS SECTION, YOU WILL: Get ideas and tips about developing architects' career paths.

KEY POINTS:

- A strong engineering background is essential for architects to make informed technology decisions and build effective relationships with developer teams.
- Moving from an engineering role to an architecture role involves broadening scope, increasing diversity, and developing strong communication and influencer skills.
- Career tracks can include Senior Architects (broader responsibilities), Principal Architects (specialized focus), and Enterprise Architects (aligning technical strategy with business objectives).
- Architecture roles can lead to tech leadership positions such as Engineering Director or Chief Technology Officer (CTO), leveraging strategic vision, decision-making, and leadership skills.

A **career path** is all about the journey you take in your professional life, reflecting the roles and responsibilities you take on along the way. In any field, you might find this path to be **linear**, with a straightforward climb through positions, or **non-linear**, where you make lateral moves, dive deeper into a specialization, or even switch to different areas altogether. Both approaches offer valuable growth opportunities—what matters most is aligning your career path with your personal goals, skills, and the environment you're in.

In this section, let's chat about the **various career trajectories** available to IT architects. Whether you're keen on becoming a technical expert or aiming for a strategic leadership role, there are plenty of avenues to explore. I'll discuss how architects can grow within their current roles, move up to leadership positions, or even pivot into related fields like product management, operations, or executive management.

If you're looking for practical tools and tips on managing, developing, or hiring architects, be sure to check out the **Appendix**—it's packed with helpful resources.

3.1: Solid Engineering Background

My perspective on architecture is rooted in a **strong engineering foundation**—and for good reason. In my experience, the most effective architects tend to have a **robust engineering background**.

Engineering foundation

image by kobus louw from istock

While there are always exceptions, architects lacking substantial hands-on experience in software development often struggle to make sound technical decisions and build credibility with engineering teams.

3.1.1: Why Engineering Experience Matters

Architects with a solid foundation in software engineering bring more than just technical knowledge—they bring **practical wisdom**. They have written code, debugged systems, managed technical debt, and experienced production outages. This firsthand experience enables them to:

- **Make realistic decisions** based on the constraints and trade-offs of real-world systems,
- **Select appropriate technologies and patterns**, not merely based on trends, but on the specific context,
- **Identify potential pitfalls and technical challenges** before they become delivery risks.

This type of experience is not theoretical—it is gained through time spent in the field. Consequently, engineers are far more likely to trust and follow the guidance of architects who have shared that journey.

3.1.2: Bridging Vision and Execution

A solid engineering foundation also enhances an architect's ability to **communicate effectively with development teams**. Architects who are fluent in the technical language and culture of engineering can bridge the gap between high-level architectural vision and the day-to-day realities of implementation. They don't just design solutions—they **collaborate on them**, guiding teams without micromanaging or over-specifying.

3.1.3: Thinking Like an Engineer

Engineers are trained to:

- **Analyze deeply**,
- **Think critically**,
- **Solve complex**, **interdependent problems**.

These skills are essential for architects, who must evaluate trade-offs, navigate ambiguity, and devise solutions that are both technically sound and aligned with business goals.

While becoming a great architect involves learning to think strategically, communicate cross-functionally, and lead with influence, the **foundation is engineering**. Without it, architecture risks becoming

disconnected from implementation—an abstract exercise with minimal practical impact.

In short: **real influence starts with real experience**. An engineering foundation not only enhances architects' credibility but also their capability.

3.2: Entering the Architecture Space

While a **strong engineering foundation** is essential, transitioning into an architecture role requires more than just technical proficiency. The shift from engineer to architect involves a significant expansion in focus, responsibilities, and skill sets. Architects work at the intersection of **technology**, **business**, **and people**, and succeeding in this role depends on adapting to that broader context.

Stepping into architecture

image by vm from istock

Here are the three key shifts that define this transition:

3.2.1: 1. Broader Scope

Engineers often concentrate on specific components or features. In contrast, architects must adopt a **system-wide perspective**. This includes:

- Considering the interactions between services, systems, and platforms,
- Understanding how different parts interact, scale, and evolve,
- Balancing performance, security, reliability, and cost across the architecture.

Architects are responsible for designing **cohesive**, **scalable solutions** that not only fit the technical landscape but also align with the **strategic and operational context** of the organization.

3.2.2: 2. Higher Diversity

The architectural role comes with **greater variation** in both the types of work and the people you collaborate with. This includes:

- A broader array of technologies and system types,
- Exposure to business processes, compliance concerns, and customer requirements,
- Collaboration with cross-functional teams—from developers and designers to executives and finance professionals.

This diversity requires versatility. Architects must be able to **context-switch**, communicate across disciplines, and apply **systems thinking** to tackle complex and often ambiguous problems.

3.2.3: 3. Changing Skillset

Perhaps the most significant change is the increased emphasis on **communication and influence**. Architects must:

- Translate technical concepts into business value,
- Communicate clearly with both technical and non-technical stakeholders,
- Build trust, **facilitate decision-making**, and align teams around a shared vision.

Soft skills—such as empathy, negotiation, and storytelling—become as crucial as technical expertise. Without these skills, even the best technical solutions may struggle to gain traction or deliver value.

3.2.4: The Role in Context

Architects must be **multifaceted professionals**—grounded in engineering, fluent in business, and skilled in stakeholder engagement. It's not enough to know how to build systems; architects must also understand **why those systems matter** and **how they support strategic goals**.

The transition into architecture is not merely a promotion—it's a **paradigm shift**. It represents a move from solving problems in code to addressing challenges in systems, teams, and organizations. Those who embrace this expanded scope will find architecture to be a uniquely challenging and rewarding discipline.

3.3: Career Progression in IT Architecture

A career in IT architecture is rarely one-size-fits-all. Instead, it's a journey that can evolve in many directions, with titles, responsibilities, and areas of focus varying greatly between organizations.

Career Path Image

image by bowie15 from istock

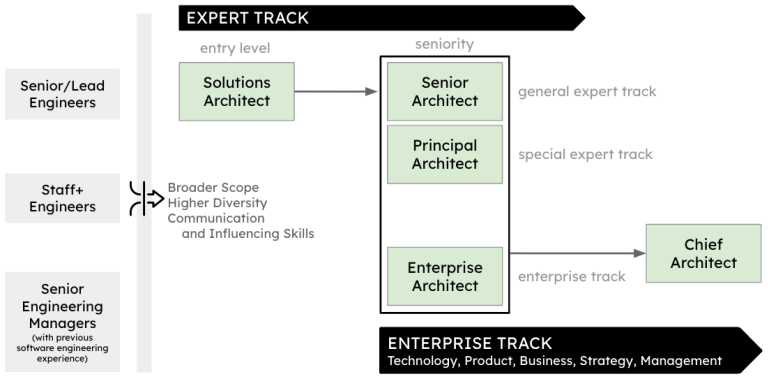
Most architects begin their careers as **hands-on solution architects**, where they design and deliver technical solutions while remaining closely connected to engineering teams. From there, career progression typically branches into one of **three key tracks** (see Figure 1):

3.3.1: Generalist Track – Senior Architect

Senior Architects follow a generalist path characterized by **breadth and adaptability**. These architects:

- Tackle complex, cross-domain problems.
- Move between high-priority areas as organizational needs shift.
- Act as integrators across technical teams.
- Maintain a holistic view of system architecture and ensure its coherence.

They are especially valuable in large, fast-changing environments where **flexibility and system-wide understanding** are critical.



Architect Career Paths

***Figure 1**: Example of career paths within IT architecture.*

3.3.2: Specialized Track – Principal Architect

Principal Architects evolve from senior roles into **deep subject-matter experts**. They:

- Own and advance a particular domain (e.g., data, distributed systems, frontend architecture).
- Provide **thought leadership** and establish a long-term vision.
- Define standards and best practices.
- Mentor engineers and other architects in their specialty.

This track is ideal for those who prefer to **go deep rather than broad** and want to shape the future of key architectural pillars.

3.3.3: Enterprise Track – Enterprise Architect

Enterprise Architects operate at the intersection of **technology and business strategy**. They:

- Collaborate closely with senior engineering and business leaders.
- Align architectural decisions with strategic goals.
- Navigate cross-functional initiatives that span departments and domains.
- Shape and communicate the long-term technology vision for the organization.

This track is suited for architects who enjoy thinking **systemically**, influencing organizational structure, and driving large-scale transformation.

3.3.4: Evolving Beyond Titles

While job titles and tracks provide helpful structure, **true career progression in architecture is about increasing impact**. The most successful architects are those who:

- Continuously learn and adapt.
- Build trust and credibility across teams.
- Drive innovation that aligns with real business needs.
- Translate abstract strategy into actionable execution.

Whether they hold a specific title or not, these architects play a **critical role in the success of their teams and organizations**. Their career path is defined less by hierarchy and more by **the value they create** and the problems they solve.

3.4: Career Progression Beyond IT Architecture

A career in IT architecture often serves as a **launchpad into senior technology leadership roles**, such as **Engineering Director**, **Head of Technology**, or **Chief Technology Officer (CTO)**. This progression is both natural and achievable, as the skills and experiences gained in architectural roles closely align with the demands of executive leadership.

Leadership Path

image by miniseries from istock

3.4.1: Why Architecture Is a Stepping Stone to Tech Leadership

Tech executives with architectural backgrounds are often **trusted by engineering teams** and respected by business leaders for their clarity

and technical judgment.

3.4.1.1: 1. Strategic Vision

Architects are trained to **connect technology to business value**. They design systems that meet current needs while preparing for future demands—an essential mindset for tech leaders. This long-term strategic thinking serves as the foundation for roles that shape **organizational direction and innovation**.

3.4.1.2: 2. Cross-Functional Influence

Architects frequently collaborate with **product managers**, **executives**, **operations**, **and finance**. These interactions provide insight into how organizations function beyond just code and infrastructure. This cross-functional exposure helps architects develop a **broad systems view**—a critical trait for leaders who must balance priorities across departments.

3.4.1.3: 3. Leadership and Mentorship

As architects advance in their careers, they often take on informal leadership roles—**mentoring engineers**, shaping engineering culture, and guiding project direction. These soft leadership skills—earned through influence rather than authority—translate seamlessly into formal leadership roles that require **team-building**, **communication**, **and motivation**.

3.4.1.4: 4. Deep Technical Expertise

Unlike many pure management tracks, architects possess a strong, credible understanding of technology. This expertise enables them to:

- Make well-informed strategic decisions,
- Evaluate technical trade-offs,
- Foster innovation through architectural foresight.

3.4.2: From Architect to Tech Executive: A Natural Evolution

The transition from architecture to tech leadership isn't just a change in title—it marks a **shift in focus** from designing systems to **designing organizations and strategies**. However, the core capabilities—**systems thinking, strategic alignment, technical depth, and influence**—remain consistent.

Many successful CTOs and Engineering Directors began their journeys as architects. Their impact at the architectural level provided a strong foundation to lead **engineering organizations, innovation portfolios, and digital transformation efforts**.

Whether you aspire to lead technology teams, drive enterprise strategy, or shape company-wide innovation, a career in IT architecture can offer a **powerful and purposeful path forward**.

3.5: To Probe Further

- Appendix: Resources for Managing, Growing, and Hiring Architects[1]
- Appendix: Architect Archetypes[2]
- Software Architect Archetypes[3], by Gergely Orosz, 2023

[1] growing
[2] archetypes
[3] https://newsletter.pragmaticengineer.com/p/software-architect-archetypes

3.6: Questions to Consider

- *Reflect on career paths in architecture. How can an engineering background impact effectiveness of an architect?*
- *Reflect on your career progression in architecture. How can you continuously stay relevant and make an impact in your role?*
- *If you were involved in the hiring process for architects, how would you assess a candidate's technical skills, communication and collaboration skills, leadership and problem-solving abilities, and cultural fit?*
- *What strategies would you implement to ensure you continuously raise the bar in developing and hiring architects in your organization?*
- *How could you demonstrate your communication and collaboration skills as an architect? Can you share an instance where these skills are crucial?*
- *How would you describe your leadership and problem-solving abilities? Can you share an example of how you've used these skills in your work?*
- *Reflect on the cultural fit between you and your organization. How do your values align with those of the company?*
- *What steps would you include in your hiring process for architects to ensure a solid evaluation of the candidates?*
- *How would you ensure diversity of perspectives within your architecture team, and is this important?*

4: Building Skills

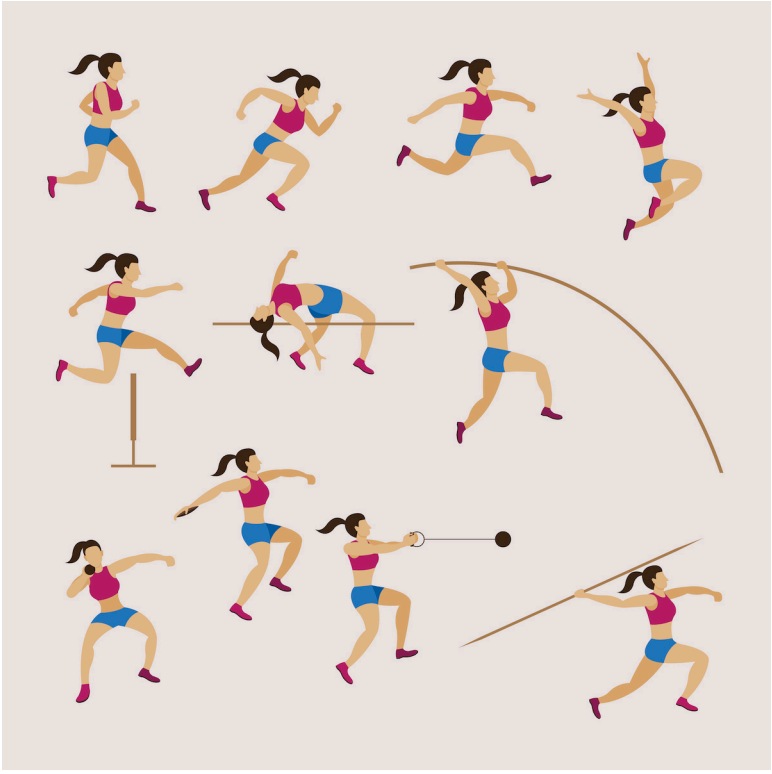


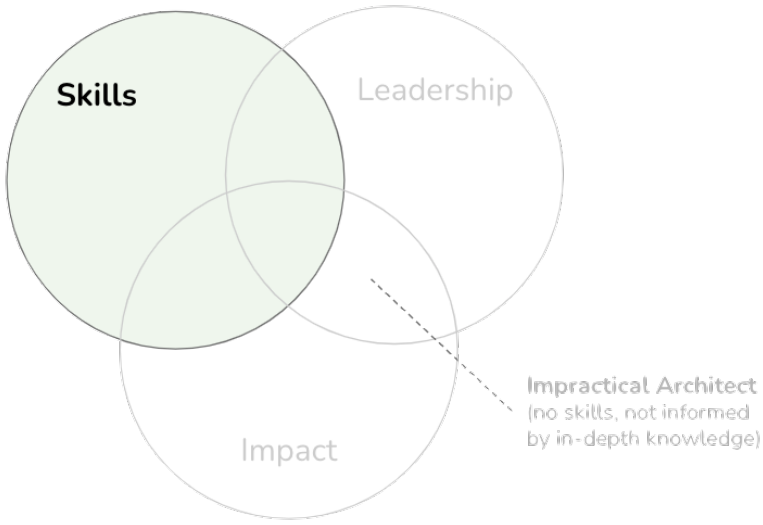
image by muchmania from istock

IN THIS SECTION, YOU WILL: Understand that architects' skills should include a mix of technical, communication, product development, and business skills, and get valuable pointers to resources for developing these skills.

KEY POINTS:

- An architect's typical skillset includes hard (technical) skills, soft (people & social) skills, product development, business skills, and decision-making skills.
- Hard (technical) skills are essential for designing, implementing, and maintaining an organization's technology landscape.
- Soft skills are integral to social architecture, enabling individuals to navigate and contribute to these social systems effectively.
- Product development knowledge is the bridge that helps architects align technical solutions with customer needs and business objectives.
- Business domain knowledge is not just useful but essential for architects to create solutions that deliver real value.
- Decision-making skills ensure that architectural decisions are sound, sustainable, and aligned with long-term strategic objectives.

Simply knowing things isn't enough. With just an internet connection, you can find information easily. What really counts is knowing **how to apply that knowledge** when faced with challenges—when requirements conflict, teams disagree, and ambitions clash with constraints. This is where architects step in (Figure 1).



Architect Skills

Figure 1: Skills play a crucial role in architectural effectiveness, alongside impact and leadership. Without them, architectural decisions can easily veer into impracticality.

4.1: Skills That Make Architecture Work

To truly succeed, IT architects need more than just technical know-how; they have to be able to **connect systems**, **people**, **and business contexts** into something cohesive and valuable.

Let's break it down.

4.1.1: Technical Skills: The Foundation

Good architecture starts with a strong technical foundation. You can't effectively guide others through a system you don't understand. This means being fluent in:

- **Modern infrastructure**—think cloud platforms, containers, and orchestration tools.
- **Legacy systems**—because systems are built on what already exists.
- **Languages**, **patterns**, **and frameworks**—these are your essential tools for design and development.

But it's not just about being the smartest person in the room. It's about having enough knowledge to make **pragmatic**, **scalable decisions**, and enough breadth to communicate effectively with different teams without losing credibility.

Great architects don't just read whitepapers; they write code, debug systems, and focus on **long-term sustainability**, rather than just short-term fixes.

4.1.2: Communication Skills: The Connector

If technical skills are the groundwork, communication is the bridge. Architects often interact with various groups: developers, product managers, executives, and even customers. Each audience has its own

way of communicating, so architects must be able to **switch gears**—not just in language but also in thinking.

This involves:

- Breaking down complex ideas into straightforward, compelling messages.
- Actively listening—truly grasping people's concerns and feedback.
- Adjusting tone and medium based on who you're talking to.

When done right, effective communication aligns teams, reduces misunderstandings, and **builds trust**—a crucial asset that amplifies your influence.

4.1.3: Influence Skills: Leading Without Authority

One thing that isn't discussed much early on is that **architects don't usually have formal power**. They can't force a team to adopt a design or mandate changes (that is, if they want those changes to stick).

Instead, architects learn to **lead through influence**. This means:

- Gaining credibility through consistent and sound judgment.
- Fostering alignment between teams, even when it's tough.
- Supporting good ideas—even if they're not the most popular.

Influence isn't about being the loudest; it's about being the most **respected**, most **steady**, and most **constructive** voice in the room.

4.1.4: Product Development: Thinking Beyond Technology

Technical excellence alone won't get you far—it's about delivering products that provide value. So architects must know how product development works—how features are defined, how customers use them, and how success is measured.

This requires:

- Understanding **Agile**, **Lean**, **and other product-focused methodologies**.
- Collaborating regularly with product managers and UX designers.
- Focusing on outcomes that are **feasible**, **valuable**, **and usable**.

The best architects don't just create systems; they help build products that users want and that the business can sustain.

4.1.5: Business Domain Knowledge: Grounding the Architecture

Context is key. A payment system differs drastically from a media platform, and a hospital EMR isn't the same as a social network. Architects who grasp the **business domain**—its drivers, constraints, and opportunities—tend to design much better systems.

This means:

- Understanding the regulatory landscape and operational realities of the industry.
- Staying updated on competitive trends and challenges.
- Linking technical initiatives to broader business goals.

Without this context, architectural decisions can risk becoming ungrounded—technically solid but lacking strategic relevance.

4.1.6: Decision-Making: Navigating the Grey Areas

At its core, architecture is about **making decisions under uncertainty**. Every choice has real consequences. Each abstraction, interface, and constraint you introduce has lasting effects.

Effective architects:

- Navigate competing trade-offs (cost, time, risk, complexity).
- Make choices in ambiguous situations, even with imperfect data.
- Keep a long-term vision in mind while also making meaningful progress.

The goal isn't to achieve perfection; it's about making progress with integrity.

4.1.7: The Balanced Profile

Let's be real: nobody is great at everything. Architects are just like everyone else—some are more technical, while others are gifted at connecting with people. What really sets the great ones apart is that they understand their strengths and where they could use a little help. They're always working towards finding that balance.

At the end of the day, architecture isn't something you do alone. It's all about teamwork, staying aware of your surroundings, and handling high-pressure situations together. The architects who truly succeed are the ones who don't just focus on building systems but also on building relationships—linking technology with people and a sense of purpose.

4.2: Building Technical Skills

Technical skills, also known as **hard skills**, are the foundational abilities that architects need to **design**, **implement**, **and maintain** an organization's technology landscape. These skills allow architects to develop robust, scalable, and maintainable systems that support business goals while ensuring performance, security, and long-term adaptability.

Technical Skills

image by rgstudio from istock

Below are some essential technical skills that architects must master to be effective in their roles:

4.2.1: System Design

The ability to **design complex systems**[1] from the ground up is fundamental. Architects should be able to envision and structure systems that balance functionality, performance, and maintainability. Strong system design skills ensure that architecture aligns with strategic objectives and scales with business growth.

4.2.2: Engineering Processes

Architects must understand modern engineering practices[2], such as:

- Agile methodologies
- DevOps and CI/CD pipelines
- Software development lifecycle (SDLC) best practices

These skills ensure that systems are built efficiently and integrated seamlessly into existing workflows.

4.2.3: Design Patterns and Tactics

Familiarity with proven architectural patterns[3]—such as **MVC**, **SOA**, **microservices**, and **cloud-native design**—enables architects to:

- Solve recurring design problems
- Promote modularity and reuse
- Ensure scalability and maintainability

Architects must also understand **design tactics**[4] for addressing quality attributes such as availability, performance, and modifiability.

[1] https://blog.pragmaticengineer.com/system-design-interview-an-insiders-guide-review/
[2] https://obren.io/tools/catalogs/?id=design-tactics-high-performing-technology-organizations
[3] https://obren.io/tools?tag=design_patterns
[4] https://obren.io/tools?tag=design_tactics

4.2.4: Security and Privacy by Design

Security[5] must be embedded in the design process, not merely added later. Architects need to:

- Integrate security practices early in the design phase
- Address privacy concerns
- Ensure compliance with regulatory standards

A proactive approach to **threat modeling**, **encryption**, and **data governance** is essential.

4.2.5: System Optimization

Performance and scalability[6] should not be afterthoughts. Architects should:

- Identify bottlenecks using profiling and monitoring tools
- Apply tuning techniques to enhance system throughput
- Design for elasticity and graceful degradation

These skills help create systems that perform well under real-world conditions.

4.2.6: Source Code Structure & Maintainability

Architects must advocate for clean, maintainable code[7] by:

- Promoting **clear module boundaries** and **sensible abstractions**
- Encouraging practices like **refactoring**, **documentation**, and **code reviews**
- Reducing technical debt over time

Good code structure lays the groundwork for long-term sustainability.

[5] https://obren.io/tools?tag=security
[6] https://obren.io/tools/catalogs/?id=design-tactics-sig-performance
[7] https://obren.io/tools/catalogs/?id=design-tactics-sig-maintainability

4.2.7: Reliability and Stability Patterns and Tactics

Architects must design systems with failure in mind[8]. This means:

- Anticipating and addressing common failure scenarios
- Implementing **redundancy**, **circuit breakers**, and **failover mechanisms**
- Ensuring graceful degradation to protect user experience

These patterns and tactics[9] are essential for **building resilient systems**.

4.2.8: Usability

While not always the focus for backend architects, **usability principles**[10] are crucial when designing systems that:

- Support internal tools or platforms
- Interface with end-users or customers
- Influence adoption and efficiency

Architects should collaborate with product and UX teams to ensure systems are **intuitive**, **accessible**, **and supportive of real-world tasks**.

4.2.9: Evolving Your Technical Skills

Technology evolves rapidly, and architects must evolve with it. Ongoing learning is essential—whether through hands-on experience, formal education, mentoring, or curated resources.

For further reading and recommended tools to build these skills, please refer to the Appendix.

[8] https://obren.io/tools/catalogs/?id=releaseit-stability-awareness
[9] https://obren.io/tools/catalogs/?id=releaseit-stability-tactics
[10] https://obren.io/tools?q=usability

4.3: Developing Soft Skills

> *"To change the architecture of a software-intensive system embedded in a large organization, you often have to change the architecture of the organization itself. Ultimately, that is a political problem, not just a technical one."*
> — Grady Booch

Soft skills—often referred to as **interpersonal**, **emotional**, **or behavioral skills**—are essential to effective architectural practice. They are a core component of what we call **social architecture**: the art of shaping relationships, influencing others, and fostering organizational alignment. Technical systems exist within social systems, and to design and evolve one, you often need to navigate and reshape the other.

Soft Skills

image by peopleimages from istock

4.3.1: Communication Skills

Clear and adaptable communication[11] is vital for architectural influence. Architects must:

- Write clearly and persuasively (written[12]),
- Visualize ideas effectively through diagrams and presentations (visual[13]),
- Speak with clarity and confidence (verbal communication),
- Listen actively to understand, not just to respond.

[11] https://obren.io/tools?tag=consultancy
[12] https://obren.io/tools/sowhat/
[13] https://obren.io/tools?tag=visuals

These skills help bridge gaps between teams, align stakeholders, and reduce misunderstandings that can derail initiatives.

4.3.2: Networking & Collaboration

Architects rarely succeed in isolation. Their impact depends on strong **relationships**[14] across both technical and non-technical domains. Key skills include:

- Building a broad, cross-functional network,
- Partnering across all levels—from engineers to executives,
- Facilitating collaboration in diverse teams with varying perspectives.

Networking opens doors, while collaboration keeps them open.

4.3.3: Organizational & Time Management Skills

Given their high levels of responsibility and often competing demands, architects must:

- Prioritize effectively,
- Set and manage realistic goals,
- Delegate tasks where appropriate,
- Stay organized across complex, multi-team engagements.

Strong organizational skills[15] lead to **better outcomes**, **reduced stress**, **and sustained effectiveness** over time.

[14] https://obren.io/tools?tag=leadership
[15] https://obren.io/tools?tag=reflect

4.3.4: Strategy & Problem Solving

Architects operate in complex environments. Their decisions must be informed, strategic, and forward-looking[16]. Key skills include:

- Breaking down complex problems and identifying root causes,
- Evaluating options and understanding long-term trade-offs,
- Driving solutions that are both technically feasible and aligned with business objectives.

Great architects combine **analytical rigor with creative problem-solving**, spotting patterns that others miss and shaping paths that others haven't yet imagined.

4.3.5: The Value of Soft Skills in Architecture

Soft skills are what **transform knowledge into meaningful impact**. They enable architects to:

- Influence without authority,
- Facilitate alignment between teams and departments,
- Build credibility, trust, and consensus across organizational silos.

They also **foster environments where innovation, collaboration, and resilience can thrive**—hallmarks of high-performing organizations.

Developing these skills isn't optional; it's a vital aspect of becoming a truly effective architect.

For additional resources on building these competencies, see the Appendix.

[16] https://obren.io/tools?tag=it

4.4: Product Development Skills

Product development is the comprehensive process of transforming an idea into a successful market offering. It includes everything from ideation and design to validation, launch, and post-release refinement. For product-led organizations, **the product serves as the engine of growth**, with all strategies focused on delivering value through it.

Architects play a crucial role in ensuring that the **technical foundation supports the product's success**. Understanding the product development lifecycle enables architects to work more effectively with product managers, designers, and other stakeholders, ensuring that technical decisions align with customer and business goals.

Product Development

image by tirachard from istock

4.4.1: Key Stages of Product Development

Here are the primary stages involved in developing successful products and their significance for architects:

4.4.1.1: Idea Generation

New product ideas often emerge from **customer feedback**, **market research**, **internal brainstorming**, **and emerging technologies**. This creative phase requires open-mindedness and a willingness to challenge assumptions.

Why it matters: Architects can support this phase by providing insights on technical feasibility early on, preventing unrealistic ideas from progressing too far down the pipeline.

4.4.1.2: Market Research

This stage involves assessing customer needs, competitive positioning, and potential market fit, ensuring that the product addresses the **right problem for the right audience**.

Why it matters: Architects gain valuable context here, which enables them to design solutions that are not only technically sound but also commercially relevant.

4.4.1.3: Product Design and Development

Translating validated ideas into tangible products involves **collaboration between design**, **engineering**, **and architecture**. Prototypes are created, tested, and refined.

Why it matters: Architects are key in defining system boundaries, selecting technologies, and ensuring that **design decisions scale and integrate well** with the broader ecosystem.

4.4.1.4: Testing and Validation

The product is tested for functionality, usability, and performance. User feedback is collected and utilized to improve the solution before launch.

Why it matters: Architects help define testability criteria, establish performance baselines, and **ensure that systems are resilient and observable**.

4.4.1.5: Marketing and Launch

A successful launch includes **branding**, **pricing**, **promotion**, **and distribution plans**. Timing and positioning are crucial.

Why it matters: Architects can assist in preparing for potential traffic surges, integrating with marketing platforms, and ensuring infrastructure readiness.

4.4.1.6: Post-Launch Evaluation and Iteration

After release, the product is monitored for performance, adoption, and user feedback. Iterations and updates are implemented to enhance functionality or address issues.

Why it matters: Architects contribute to developing systems that are easy to evolve and monitor, enabling **continuous improvement** without compromising stability.

4.4.2: Why Product Development Skills Matter for Architects

Modern architects do not work in isolation; they function within product ecosystems. Understanding the **full lifecycle of product development** allows architects to:

- Collaborate more effectively across functions,
- Anticipate business trade-offs,
- Ensure their designs are **not only technically sound but also strategically aligned**,
- Influence roadmaps by highlighting opportunities and constraints early on.

Ultimately, architects who understand the product mindset help create **solutions that are desirable, viable, and feasible**—the cornerstone of successful product delivery.

For resources to enhance your product development skills, see the Appendix.

4.5: Business Skills

Regardless of their technical depth or design expertise, **architects must understand how businesses operate**. Their ability to make meaningful contributions relies not only on system architecture but also on understanding **how technology decisions support strategic goals**.

Great architecture is not just technically sound; it is also **aligned with business value**.

Business Skills

image by azmanl from istock

4.5.1: Core Business Skills for Architects

Here are the essential business-related competencies that architects must develop to be effective collaborators and decision-makers:

4.5.1.1: General Business Knowledge

Architects benefit from a solid understanding of foundational business concepts, including:

- Finance and budgeting
- Marketing and sales dynamics
- Operations and supply chains
- Strategic planning

Understanding these domains helps architects **see the big picture** and communicate more effectively with stakeholders across the organization. A great resource to start with is *The Personal MBA*[17]—a highly accessible guide to essential business thinking.

4.5.1.2: Specific Business Domain Expertise

Beyond general knowledge, architects should immerse themselves in the **specific domain**[18] their organization operates in—whether it's healthcare, finance, manufacturing, retail, or education. This includes knowledge of:

- Industry regulations
- Market trends
- Competitive forces
- Customer expectations

Domain expertise enables architects to design systems that **solve the right problems** and support key business priorities with precision.

[17] https://personalmba.com/
[18] https://obren.io/tools?tag=domain_models

4.5.1.3: Business Analysis & Requirements Gathering

Effective architects must be skilled at:

- Interpreting business needs
- Facilitating workshops and stakeholder interviews
- Translating business goals into **clear**, **actionable technical requirements**

Techniques like **SWOT analysis**, **value stream mapping**, and the **business model canvas** can help frame problems and identify high-value opportunities.

4.5.1.4: Stakeholder Management

Architects frequently act as a bridge between technical and non-technical stakeholders. This requires:

- Building trust with executives and managers
- Understanding stakeholder concerns
- Navigating competing priorities with empathy and clarity

Strong relationships with stakeholders help architects **gain buy-in and drive alignment**.

4.5.1.5: Project Management Awareness

While architects aren't always project managers, understanding project management fundamentals—such as scope, time, cost, and risk—helps them:

- Communicate more effectively with project leads
- Contribute to realistic planning
- Ensure that architectural decisions support **deliverability**

Familiarity with agile and hybrid methodologies is especially valuable.

4.5.1.6: Financial Acumen

Architects should be knowledgeable about key financial concepts, including:

- **Budgeting and forecasting**
- **ROI (Return on Investment)**
- **TCO (Total Cost of Ownership)**
- **EBITDA** and other financial KPIs

This knowledge allows architects to evaluate solutions not just for **technical merit** but also for **economic impact**—a crucial skill in resource-constrained environments.

4.5.1.7: Strategic Thinking

Architects must understand how their work fits into the **long-term business vision**. This includes:

- Identifying innovation opportunities
- Aligning architectural decisions with growth objectives
- Recognizing technology's role in **competitive differentiation**

Strategic architects help guide the organization rather than simply serve it.

4.5.1.8: Change Management

Architectural changes often trigger organizational changes. Architects should be familiar with:

- Change readiness assessments
- Communication planning
- Resistance management strategies

These skills support **smooth transitions** when introducing new platforms, tools, or structures.

4.5.2: The Value of Business Fluency in Architecture

Architects who understand the business landscape can:

- Speak the language of executives
- Align systems with strategic priorities
- Anticipate risks beyond the technical domain
- Influence high-level decisions

This **business-savvy mindset** transforms architects from technical advisors into **strategic partners**, enabling organizations to realize the full value of their technology investments.

For further reading and training materials, see the Appendix.

4.6: Decision-Making Skills

In architecture, **decision-making is a core function**, not a secondary responsibility. Strategic decisions shape organizations, and architects who are not involved in these decisions will have **limited influence and impact**.

Architects play three roles in decision-making:

1. **Decision-Makers**: They own the technical and strategic choices that guide architectural outcomes.
2. **Advisors**: They support others (e.g., engineering leads, executives) in making informed decisions.
3. **Evaluators**: They assess existing decisions to provide feedback or suggest course corrections.

Architects must act as **navigators** who guide initiatives through uncertainty, risk, and opportunity. Their effectiveness relies on their ability to **evaluate trade-offs**, balance priorities, and facilitate alignment between technical and business domains.

Decision-Making

image by gorodenkoff from istock

4.6.1: Core Decision-Making Competencies

Here are essential skills and mental models that architects should cultivate:

4.6.1.1: Decisions = Irrevocable Resource Allocations

Every decision commits finite resources: time, people, money, infrastructure, and opportunity cost. Architects must make decisions **intentionally**, understanding the **long-term trade-offs** and systemic consequences.

4.6.1.2: Avoid the Outcome Bias

Outcome bias occurs when individuals evaluate decisions based on results rather than the soundness of the decision-making process. Architects need to focus on **process quality**, not hindsight. A good decision can lead to a bad outcome—and vice versa.

4.6.1.3: Know When to Trust Intuition

Not all decisions require exhaustive analysis. In ambiguous or fast-moving contexts, **intuition shaped by experience** can be valuable. Architects should recognize when it's appropriate to trust their instincts—and when it is not.

4.6.1.4: "No Decision" Is Still a Decision

Deferring a choice is itself a decision, with associated opportunity costs. Architects must understand that **inaction creates risks**, often more significant than committing to an imperfect path and adjusting later.

4.6.1.5: Risk Assessment

Architects must identify and evaluate:

- Technical risks

- Business impact risks
- Organizational risks

Balancing **innovation with resilience** is crucial for making well-rounded decisions.

4.6.1.6: Collaborative Decision-Making

Involve the right people. Architects should:

- Seek diverse input.
- Facilitate consensus when needed.
- Clarify accountability and ownership.

Collaborative decisions garner more support and reduce resistance.

4.6.1.7: Ethical Considerations

All architectural decisions have consequences—for users, employees, and ecosystems. Architects should weigh:

- Privacy and security
- Accessibility
- Long-term societal impact

Doing the **right thing** is just as critical as doing things efficiently.

4.6.1.8: Adaptability & Reversibility

Good architects know when to **hold firm** and when to **change direction**. Utilize reversible decisions for experimentation and commit deeply only when the path is clear. Design systems—and strategies—that leave **room to pivot**.

4.6.1.9: Leverage Decision Intelligence

Decision intelligence integrates data, models, and human judgment, enabling architects to move from "best guesses" to **evidence-based actions** through:

- Metrics
- Forecasting
- Scenario planning
- Feedback loops

4.6.2: Architects as Strategic Guides

Mastering decision-making transforms architects into **trusted strategic advisors**. This enhances their ability to:

- Champion scalable, value-aligned solutions.
- Influence roadmaps and leadership thinking.
- Protect the organization from short-sighted or reactive choices.

Ultimately, **great architecture is the byproduct of great decisions**—rooted in context, guided by insight, and implemented with clarity.

4.7: Integrating Skills for Success

To make a real impact, architects need to **bring together a variety of skills** in a cohesive and strategic manner. It's not just about being brilliant on the technical side or having top-notch communication skills; the magic happens when you **blend technical expertise, interpersonal skills, and strategic thinking**.

Image

image by colorsandia from istock

Another key aspect is having a mindset of **lifelong learning**. In our fast-paced world, staying static just won't cut it. The best architects are those who adapt and grow, keeping up with new challenges and trends. They aren't just experts; they're innovative thinkers who evolve along with the systems and the people they work with.

So, how do all these skills play out in real life? Let's break it down:

- **Technical Proficiency**: This is what lets architects create solid, scalable, and future-ready systems.

- **Effective Communication**: It's crucial that everyone—whether they're tech-savvy or not—understands the architect's vision.
- **Influence and Persuasion**: These skills are key to getting everyone on the same page and moving architectural initiatives forward.
- **Product Insight**: This helps connect the dots between technical solutions and what real customers actually need, ensuring they're relevant in the market.
- **Business Understanding**: Architects need to tie their decisions back to what the organization is ultimately trying to achieve.
- **Decisive Leadership**: When challenges arise, strong architects know how to navigate uncertainty and conflicting priorities with confidence.

By weaving these capabilities together, architects become **strategic connectors**. They can seamlessly operate at the intersection of business goals, user needs, and technical realities. This balanced skill set not only boosts their effectiveness but also transforms them into **trusted leaders**. They guide organizations through complex challenges with clarity, cohesion, and a real sense of impact.

4.8: To Probe Further

- Appendix: Bookshelf[19]
- Old Books that Every Architect Should Read[20], by Gregor Hohpe, 2024
- Back from the engine room[21], by Gregor Hohpe, 2023
- Debugging Architects[22], by Gregor Hohpe, 2021

[19] bookshelf
[20] https://architectelevator.com/architecture/classic-architecture-books/
[21] https://architectelevator.com/transformation/debugging-architect/
[22] https://architectelevator.com/architecture/engine-room/

4.9: Questions to Consider

- *On a scale from 1 to 10, how would you rate your current architectural skill sets, considering technical, communication, product, business skills, and decision-making skills?*
- *Reflect on your technical skills. How proficient are you in system design, understanding engineering processes, recognizing design patterns and tactics, ensuring security and privacy, optimizing systems, and maintaining code structures?*
- *Do you need to develop specific hard skills to enhance your architectural performance?*
- *How effectively do you communicate (in writing, visually, verbally, and through listening)? How strong are your networking and collaboration skills, and how well do you manage your time and organizational tasks?*
- *Can you identify an instance where your problem-solving skills and strategic thinking have significantly influenced your work as an architect?*
- *Looking at business skills, how well do you understand general business concepts, and how familiar are you with the specific business domain of your organization?*
- *How competent are you in business analysis and requirements gathering? Can you share an example where you effectively translated business objectives into functional and technical specifications?*
- *Are there any soft or business skills you need to develop or improve to succeed in your role as an architect?*
- *Reflect on how you have used your soft skills to effect organizational change. Are there areas or situations where you could have applied these skills more effectively?*
- *How do you balance developing and maintaining your hard, soft, and business skills? Is there a particular area you tend to focus on more, and why?*

5: Making Impact

image by tuiphotoengineer from istock

IN THIS SECTION, YOU WILL: Understand that architects' work is evaluated based on their impact on the organization and get guidelines for making an impact.

KEY POINTS:

- Architects' work is evaluated based on their impact on the organization.
- Architects can make an impact via three pillars: Big-Picture Thinking, Execution, and Leveling-Up.

When we think about what architects do—whether in tech, systems, or product design—it goes beyond their technical skills. What truly sets great architects apart is the **impact** they have on their organizations.

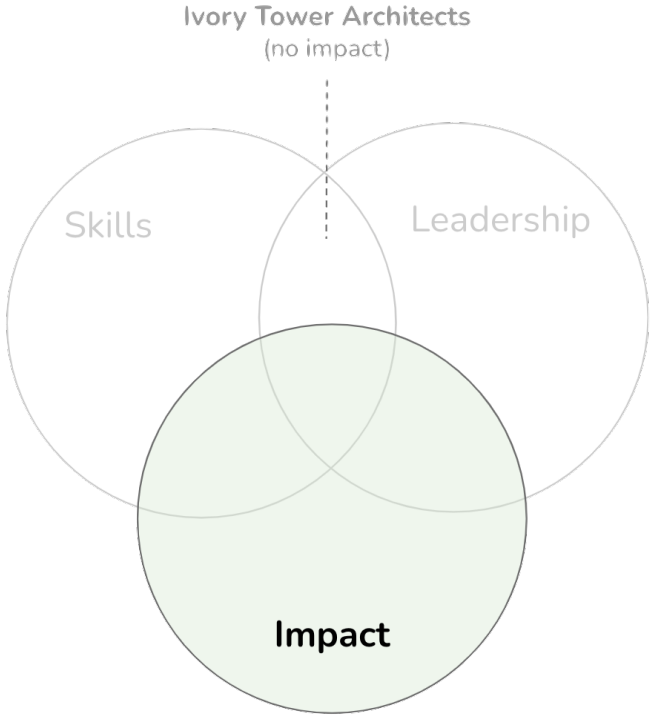


***Figure 1**: Impact is one of the three main elements of being an architect (skills, impact, leadership). Leadership without impact lacks foundation and may signal that you have become an ivory tower architect with a weak relation to reality.*

5.1: So, how can we actually measure that impact?

Architects play a crucial role in shaping outcomes, often in subtle but profound ways. Their contributions can typically be grouped into three key areas:

5.1.1: 1. Spotting and Solving Strategic Problems

Great architects don't just react to problems as they arise; they're always on the lookout for potential challenges. They have a knack for identifying high-stakes issues that could hinder growth or create inefficiencies down the line. Whether it's tackling architectural debt, addressing technical bottlenecks, or ensuring that systems align with business goals, they help lay the **groundwork for sustained success**. Their solutions are proactive and strategic, making sure that today's choices pave the way for tomorrow's achievements.

5.1.2: 2. Creating Deep and Wide Organizational Impact

Architects juggle **depth and breadth** like few others can. They dive deep into complex issues within a system or product, providing clarity and expertise where it's most needed. But they don't stop there—they also take a step back to see how those solutions can be scaled across teams and domains. This ability to **connect the dots across silos** helps transform isolated wins into broader organizational success.

5.1.3: 3. Delivering What Others Can't

Some challenges require a unique mix of vision, influence, and technical know-how, and this is when architects really show their mettle. Whether it's managing a high-stakes migration, balancing conflicting stakeholder needs, or charting a new architectural path, architects step

up when the stakes are high and the margin for error is slim. Their real value lies in their ability to **gain traction where others struggle**, driving momentum at critical moments.

In a nutshell, architects aren't just about designing systems—they're **outcome enablers**. Their success isn't just reflected in the diagrams they create or the code they review; it's in the progress they facilitate. When they tackle real business challenges, harmonize teams, and deliver lasting solutions, they make a measurable and significant difference.

5.2: Pillars of Impact

As architects, we need to cultivate not just solid technical know-how, but also strong interpersonal and business skills. Ideally, these skills get refined through real-world experiences. But as we grow in our careers, it's important to shift our focus from merely accumulating skills to actively building competencies that create real, measurable impact within our organizations.

I often find that grounding our development in **concrete activities**—real challenges that require both action and reflection—makes all the difference. This hands-on, outcome-driven approach allows us to align skill-building with actually **delivering value** in meaningful ways.

One way to frame this development journey is by looking at the Staff Engineering path. For example, Tanya Reilly's *The Staff Engineer's Path*[1] and Will Larson's *Staff Engineer: Leadership beyond the management track*[2] provide fantastic insights into the responsibilities and expectations we face in architectural roles.

[1] https://www.oreilly.com/library/view/the-staff-engineers/9781098118723/
[2] https://staffeng.com/guides/staff-archetypes/

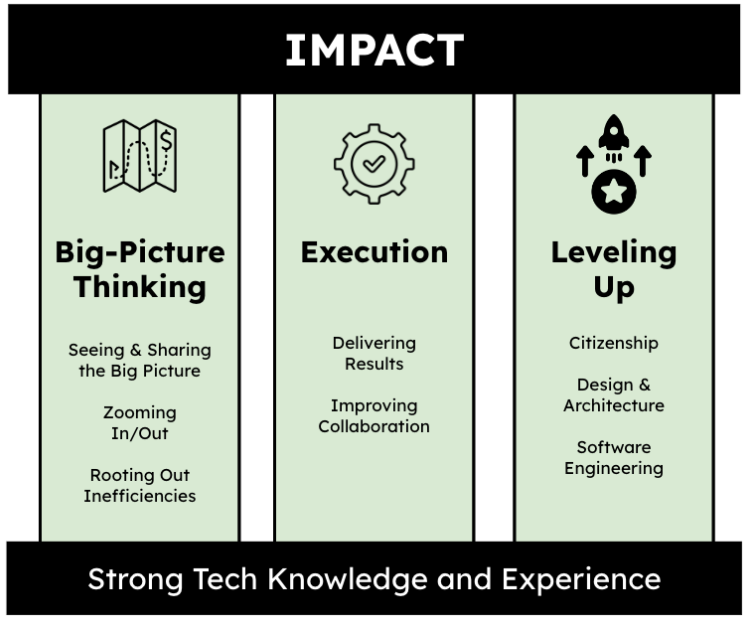


Figure 2: *Key competencies of architects. Inspired by* The Staff Engineer's Path *by Tanya Reilly.*

Based on these frameworks, I like to break down the key competencies that lead to impactful architecture into three main pillars:

5.2.1: 1. Big-Picture Thinking

First up, big-picture thinking. It's essential for architects to look beyond just the code. We need to understand how our architectural choices fit into the larger strategy of the organization, the market landscape, and evolving technologies. It's about crafting a technical vision aligned with business goals and helping everyone see that vision, too.

This includes skills like:

- Strategic planning

- Systems thinking
- Translating business needs into technical direction
- Anticipating long-term impacts and opportunities

5.2.2: 2. Execution

Next is execution. This is where our ideas translate into real outcomes. As architects, we need to help teams deliver effective and sustainable solutions, which involves managing complexity, aligning stakeholders, and guiding teams through the ups and downs of real-life delivery.

Key skills in this area include:

- Technical leadership during delivery
- Cross-team collaboration
- Risk mitigation and adaptive planning
- Ensuring quality and fit-for-purpose outcomes

5.2.3: 3. Leveling Up

Finally, let's talk about leveling up. We don't just build systems; we also build **people** and **capabilities**. Leveling up is all about continuous improvement—growing your own skills while also lifting others up. This creates resilient, high-performing teams through mentorship, coaching, and knowledge sharing.

This pillar involves:

- Coaching and mentoring
- Fostering a culture of learning
- Capturing knowledge into reusable patterns and practices
- Staying up-to-date with industry trends and techniques

By honing in on these three pillars—big-picture thinking, execution, and leveling up—we can evolve from skilled technicians into **strategic change agents**. This model not only enhances individual performance but also magnifies your influence throughout the organization. Let's embrace this intentional development path together!

5.3: Big-Picture Thinking

You know, architects in an organization often have this unique ability to maintain a "**helicopter view**." It's a bit of a rare skill that lets them see across different teams, systems, and strategies, helping them to anticipate how decisions will play out in the long run.

image by guvendemir from istock

This strategic perspective enables architects to make contributions in four key ways:

5.3.1: Spotting High-Leverage Points

Big-picture thinkers have a knack for identifying where **small changes can lead to big results**. Whether it's addressing a bottleneck in a system, fixing a misalignment between teams, or uncovering a hidden scaling issue, architects shine when focusing on areas that provide the **greatest return for the organization**.

5.3.2: Helping Others See the Whole Picture

But it's not just about keeping that big picture to themselves. Architects are great at **helping others see it too!** They share their insights through architectural reviews, roadmaps, or tools like Lightweight Architecture Analytics. This way, they boost the strategic literacy of the whole organization, creating alignment, reducing friction, and building a shared sense of purpose among teams.

5.3.3: Zooming In and Out

One of the coolest things about architects is their ability to thrive at both **high altitudes and ground level**. They can explain how a project aligns with long-term business goals, and then dive into the nitty-gritty details to ensure those ideas are practical and achievable. This talent for **bridging strategy and execution** really sets effective architecture leaders apart.

5.3.4: Uncovering Inefficiencies

From their high vantage point, architects can pinpoint **systemic inefficiencies**—like overlapping systems, redundant efforts, or misaligned dependencies—and drive improvements that benefit multiple teams. This often means suggesting better tools, simplifying architectures, or promoting reuse across different domains.

When architects consistently apply this big-picture lens, they **bring strategic clarity** to everything they engage with. They help teams not just build the right systems, but to do it in the **right way for the right reasons**.

5.4: Execution

Architects are often celebrated for their strategic vision, but they wear many hats—it's not just about creating stunning designs; they also have a crucial role in **driving real-world outcomes**. To truly make an impact, architects need to go beyond crafting elegant systems. They must focus on **delivering practical results** and fostering teamwork. Success lies in blending deep technical know-how with **a hands-on, pragmatic approach** and a commitment to **collaboration across the organization**.

Image

image by laylabird from istock

Let's dive into how architects can shine when it comes to execution:

5.4.1: Delivering Results with Pragmatism

Architects truly add value not just through their ideas but by their knack for **turning those ideas into tangible action**. This means thinking practically, understanding constraints, and keeping a sharp eye on outcomes.

- **Build Solutions That Work in the Real World**
  Instead of getting lost in theoretical designs or overly complex models, effective architects focus on creating solutions that are **doable, valuable, and grounded in reality**. They assess what's really needed and what's feasible within the organization's environment—be it technical, cultural, or operational.
- **Simplify the Complex**
  One of the architect's standout abilities is breaking down big, complicated problems into bite-sized, manageable pieces. This helps teams make steady progress, stay focused, and keep from feeling overwhelmed. Plus, it ensures that each step is practical and achievable.
- **Plan with Constraints in Mind**
  Let's be honest: ideal solutions don't always fit within real-world limits. The best architects excel at **balancing ambition with feasibility**. They consider everything from technical debt to staffing to timelines. By planning this way, they set the stage for better execution grounded in what's realistically deliverable.

5.4.2: Enhancing Collaboration

Architects are often in a unique spot where they can **connect different parts of the organization**. By building relationships, promoting alignment, and encouraging open communication, they help teams move faster and with clearer direction.

- **Create Strategic Alignment**
  A big part of an architect's role is to **clarify the bigger goals** and help everyone see how their work contributes. When teams

are aligned on purpose and direction, it cuts down on confusion, reduces wasted effort, and speeds up progress.

- **Facilitate Stronger Teamwork**
  Think of architects as **connectors and facilitators**. They often take the lead in coordinating across teams, resolving friction points, and ensuring communication flows smoothly. This helps foster better cooperation and quicker decision-making.
- **Engage Across Departments**
  Great architects don't stick to their own lanes—they **actively build bridges between teams and disciplines**. They listen to diverse perspectives, understand different needs, and weave these insights into cohesive plans. This approach builds trust and helps break down silos, ultimately boosting organizational agility.

In essence, architects who focus on execution have the unique ability to **transform big ideas into real impact**. They combine **technical expertise**, **practical judgment**, **and cross-team collaboration** to ensure that solutions aren't just smart—they're successful. By doing this, they help the entire organization work more efficiently, effectively, and in sync.

5.5: Leveling Up

When we think about architects, it's easy to see them as just technical whizzes. But really, they're much more than that! They often take on roles as **leaders, mentors, and role models** who help elevate the level of excellence in an organization. They shape not just **how systems are built** but significantly influence **how we work together and grow as a team**.

image by sanjeri from istock

Let's explore this vital leadership role through three key perspectives: **citizenship**, **design and architecture**, and **software engineering**.

5.5.1: Citizenship: Supporting the Community and Culture

Architects often act as the champions of their organization's **technical culture** and its professional community. Their impact goes far beyond just coding or design reviews.

- **Sharing Knowledge**
  Great architects don't keep their insights to themselves. Whether through internal talks, blog posts, or speaking at conferences, they share their expertise, helping others learn and bringing fresh ideas back to the team.
- **Leading by Example**
  Real leadership means tackling tough and meaningful challenges, especially those that involve collaboration across teams. When architects step up, they inspire others and show a commitment to ongoing improvement.
- **Shaping a Strong Engineering Culture**
  By demonstrating curiosity, humility, and a love for craftsmanship, architects create a workplace where learning and collaboration are valued. They help establish an environment where teams can truly thrive and do their best work.
- **Expanding Industry Influence**
  Some architects make waves beyond their own organizations. By sharing valuable insights and best practices with the broader tech community, they really help elevate industry standards.

5.5.2: Design and Architecture: Defining What "Good" Looks Like

Architects play a crucial role in establishing the **technical foundation and direction** of systems.

- **Setting and Evolving Standards**
  Thanks to their extensive experience, architects get to define and continuously refine what "good" architecture looks like. They establish guidelines and best practices that guide teams in making better design choices.
- **Solving Big-Picture Problems**
  Architects have a knack for spotting systemic issues, like challenges in scalability or integration. They craft robust, forward-thinking solutions that ensure our software can adapt and grow over time.

5.5.3: Software Engineering: Staying Hands-On and Leading Technically

Despite their strategic roles, architects don't shy away from **hands-on engineering**. Their deep expertise helps them promote technical excellence in practical, meaningful ways.

- **Demonstrating Best Practices**
  Architects set the standard in areas like clean coding, testing, documentation, and performance monitoring. By modeling high standards, they influence the habits and expectations of the entire team.
- **Tackling Tough Challenges**
  When complex technical issues come up—be it due to high scale, complexity, or something new—architects are often the first to step in and lead the charge. Their ability to tackle these challenges helps unblock teams and enhances everyone's skills.

By excelling in these areas, architects help their organizations **level up—both technically and culturally**. They raise standards, support better decision-making, and nurture a healthier, more effective engineering environment. Most importantly, they contribute to the growth and success of those around them!

5.6: Final Thoughts

You know, architecture is so much more than just technology. It's really about making a **lasting impact**. The best architects are those who look beyond diagrams and technical choices to the **real outcomes** they can create: fostering collaboration, developing smarter systems, building stronger teams, and crafting more resilient organizations.

In this article, we dove into how architects add value through **strategic problem-solving**, **influencing across teams**, **and delivering results that truly matter**. We broke down this impact into three main pillars—**big-picture thinking**, **execution**, **and leveling up**—each one playing a crucial role in making a meaningful contribution to the organizations we work with.

What really ties everything together is this idea: being an architect means showing up **with intention**. It's not just about what you know; it's about **how you apply that knowledge** and who you bring along for the journey. Whether you're helping your teams navigate through complexity, linking strategy to execution, or mentoring others, the influence you have can create ripples of change.

So let's challenge ourselves to not only build better systems but to be the kind of architects who uplift everyone around us. Let's keep pushing for growth, both for ourselves and for our teams. After all, that's what truly makes a difference!

5.7: Questions to Consider

- *Can you identify instances where you had to go deep into a specific issue and others where you needed a broad perspective across multiple teams? How did you manage both scenarios?*
- *How have you used your technical, strategic, execution, and people skills to deliver solutions? Can you share an example?*
- *How can you build on your technical, people, and business skills to positively impact your organization's performance? How do you measure this impact?*
- *As an architect, how can you develop your big-picture thinking ability? Can you give an example of how your big-picture thinking helped to identify a high leverage point for maximum impact?*
- *Reflect on your role in execution. How can you help in delivering results and improving collaboration? Can you share an example where your pragmatism resulted in a meaningful solution?*
- *What initiatives could you have taken to improve collaboration and build trust within your organization?*
- *Have you contributed to the broader technical community through tech talks, education, publications, open-source projects, etc.?*
- *How could you help solve significant problems in your area and raise the bar of the engineering culture across the company?*
- *Can you provide an example of a systemic architectural problem you identified and the solution you proposed?*
- *How would you promote and demonstrate best-in-class practices in coding, documentation, testing, and monitoring?*

6: A Framework for Architecture Leadership: Empowering with Insight and Influence

image by niserin from istock

IN THIS SECTION, YOU WILL: Understand how to apply ideas from David Marquet's work and Netflix's valued behaviors to develop

architects' leadership traits.

KEY POINTS:

- My view of architecture leadership is inspired by David Marquet's work and Netflix's valued behaviors.
- Marquet focused on leadership and organizational management, particularly emphasizing the principles of Intent-Based Leadership.
- Borrowing from Netflix's original values, I see the following behavioral traits as crucial for architects: communication, judgment, impact, inclusion, selflessness, courage, integrity, curiosity, innovation, and passion.

"*A leader is anyone who takes **responsibility** for **recognizing** the **potential in people and ideas**, and has the **courage** to develop that potential.*" —Brené Brown

When I think about architecture leadership, two key influences really stand out for me: David Marquet's leadership principles and the Netflix Culture Memo. Marquet's insights from his books, *Turn the Ship Around!* and *Leadership Is Language*, really hit home.

He emphasizes some key ideas:

- **Empowering everyone**: It's about giving people at every level the authority to make decisions.
- **Clarity of intent**: Clearly communicating what's expected can really help everyone stay aligned.
- **Decentralized decision-making**: Letting teams make their own calls fosters ownership and accountability.
- **Servant leadership**: Supporting others in their growth is crucial.
- **Continuous improvement**: Always looking for ways to get better keeps innovation alive.

On the flip side, Netflix brings a fresh perspective with its **Culture Memo**. It lays out practical ways to nurture leadership traits across the

board, not just for those in management roles. They promote values like curiosity, good judgment, courage, and the importance of speaking up to help each other grow.

Combining Marquet's and Netflix's approaches gives us a solid, modern framework for leadership that really fits the needs of today's IT architecture. It's all about **cross-functional influence**, finding **clarity in complex situations**, and leading effectively **without relying solely on a hierarchy**.

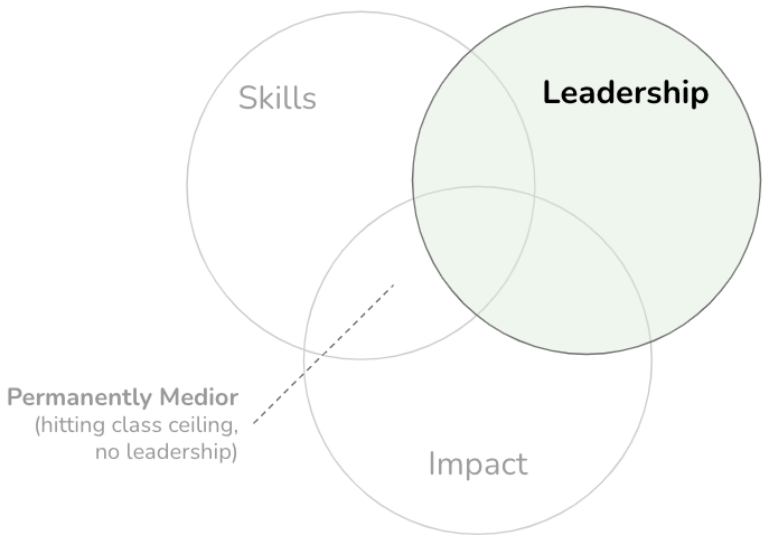


***Figure 1**: Architecture rests on three key pillars: skills, impact, and leadership. Without that leadership element, even the most skilled architects can feel stuck, struggling to scale their influence or drive meaningful change within their organizations.*

6.1: David Marquet's Work: The Leader-Leader Model

One of the most impactful leadership concepts I've come across is **David Marquet's Leader-Leader approach**. He dives deep into this idea in his books, *Turn the Ship Around!* and *Leadership Is Language*. What really sets this model apart is how it flips traditional leadership on its head. Instead of authority resting solely at the top, the Leader-Leader model encourages a **shared sense of responsibility** and **invites initiative** from everyone on the team.

At its essence, Marquet's model is built on a powerful belief: **everyone has the potential to be a leader**. Rather than relying on a single person to call all the shots, the team collaborates and shares authority to reach common goals. This creates a culture where individuals feel empowered to take ownership of their work, speak up, and act with confidence.

image by caiaimage/martin barraud from istock

For architects, this means stepping away from the habit of being the "go-to person" for every decision. Instead, they can become **facilitators**, **coaches**, **and mentors**—empowering their teams while still guiding overall architectural direction. This shift not only builds the team's capacity but also fosters trust and strengthens execution.

6.1.1: Building a Collaborative Culture

One of the standout features of the Leader-Leader model is how it nurtures **inclusion**, **accountability**, **and creativity**. When everyone is encouraged to contribute their unique perspectives and skills, teams become more collaborative and resilient. Ownership is shared, and success is celebrated as a collective achievement.

6.1.2: "I Intend To": A Small Phrase with Big Impact

In *Leadership Is Language*, Marquet emphasizes that the language we use matters a lot in leadership. He introduces the phrase "**I intend to**" as a replacement for passive, permission-seeking language. Instead of saying, *"Can I do this?"*, team members can confidently assert, *"I intend to do this."* This shift helps clarify intent and empowers individuals to take initiative.

This simple phrase can transform team dynamics in a big way.

> "**I have found the phrase 'I intend to' to be a powerful catalyst for positioning architecture work.**"

LEADER	LEADER
7. I've been doing...	7. What have you been doing?
6. I've done...	6. What have you done?
5. I intend to...	5. What do you intend?
4. I would like to...	4. What would you like to do?
3. I think...	3. What do you think?
2. I see...	2. What do you see?
1. Tell me what to do.	1. I'll tell you what to do.
WORKER	BOSS

Figure 2: *Leadership language, based on Intent-Based Leadership by David Marquet.*

When architects adopt this language themselves and encourage their teams to do the same, they help cultivate a culture rich in **clarity, action, and mutual accountability**. It sets the expectation that everyone is expected to lead, while architects are there to guide, support, and elevate the team's efforts.

6.1.3: Applying Leader-Leader Thinking in Architecture and Engineering

The Leader-Leader model isn't just a theoretical framework—it works wonders in real-world engineering settings. Here are some concrete examples where this approach shines:

- **Empowering Teams in Agile Environments**
  In Agile teams, everyone has a seat at the table when it comes to decision-making. Developers, testers, and product owners collaborate to contribute meaningfully, rather than simply following orders.

- **Cross-Functional DevOps Collaboration**
  DevOps fosters a culture of ownership throughout the product lifecycle. Teams manage everything from development to deployment, which aligns perfectly with the Leader-Leader philosophy.
- **Driving Continuous Improvement**
  DevOps also emphasizes learning through iteration. Instead of leaders controlling every decision, they coach their teams to improve through feedback loops, retrospectives, and experimentation.
- **Mentorship as Leadership**
  Senior architects can take on a mentoring role, guiding junior members through architectural challenges while empowering them to make their own decisions and grow in confidence.
- **Community-Driven Leadership in Open Source**
  Open-source projects demonstrate that leadership can be earned through contribution, not titles. Teams collaborate openly, review code together, and make decisions as a unified group—another great example of distributed authority.
- **Autonomy in Microservices Architecture**
  Teams have ownership of their services from design to deployment and support. They can make decisions independently but work together for overall success.
- **"You Build It, You Run It" Service Ownership**
  Teams that adopt this philosophy take responsibility for the long-term quality and reliability of their work, which fosters pride in ownership and motivates ongoing improvement.

6.1.4: Leader-Leader Model: Final Thoughts

The **Leader-Leader model offers a compelling framework** for architects who want to lead not by dictating but through clarity, coaching, and empowerment. It transforms the architect's role into one of **amplifying others' capabilities**, enabling the organization to move faster, innovate more, and grow stronger together.

In this vibrant environment, architects aren't just decision-makers—they're leaders nurturing their teams to take initiative and thrive.

6.2: Netflix's Valued Behaviors: A Guide for Architecture Leadership

The Netflix Culture Memo[1] is a huge influence on how I coach and develop architects. While David Marquet offers the mindset of empowering leadership, Netflix provides a behavioral blueprint that's like a practical guide for shaping strong, trusted leaders at all levels of an organization.

Their core values—**judgment**, **curiosity**, **courage**, and **selflessness**—help architects not just exemplify leadership but also nurture it in others. This is especially crucial in the field of architecture, where collaboration often trumps authority, and having a good influence is key to making progress.

image by chaiyon021/martin barraud from istock

Here's a rundown of these leadership behaviors inspired by Netflix's values, reordered for how they most directly relate to the work of IT

[1] https://jobs.netflix.com/culture

architects.

6.2.1: Communication

At the heart of effective architecture is clear and thoughtful communication. Architects thrive when they can break down complex ideas, actively listen, and adapt to different audiences.

- Be **concise and articulate** in both writing and speaking.
- **Listen actively** and aim to understand before jumping in with your thoughts.
- Keep your cool under pressure—this fosters clarity and calm.
- Tailor your communication style for multicultural and multilingual settings.

6.2.2: Judgment

Architects often step in when things get tricky or decisions are hard to make. Having sound judgment is absolutely essential.

- Use **data** to back up your intuition, blending analysis with your experience.
- Make wise decisions even when the situation is a bit fuzzy.
- Dig deep to identify **root causes** instead of just scratching the surface.
- Think about the **long-term impact** rather than chasing short-term gains.
- Be **strategic** in your thinking and communicate your priorities clearly.

6.2.3: Impact

As I often say, architecture needs to deliver real business value—not just pretty designs.

- Focus on delivering **substantial**, **high-value work**.
- Raise the performance of those around you consistently.
- Always prioritize **outcomes** over the process.
- Your work should be **reliable and trusted** by both peers and leaders.

6.2.4: Inclusion

Good architects lead across various teams, disciplines, and cultures. Making everyone feel included helps foster collaboration and leads to better decisions.

- Team up with people from **different backgrounds and experiences**.
- Embrace diverse viewpoints to strengthen your ideas.
- Evaluate others based on **talent and values**, not just similarities.
- Stay **curious about how different backgrounds shape work**, instead of brushing those differences aside.

6.2.5: Selflessness

Being a leader in architecture often means putting aside your ego and prioritizing what's best for the organization.

- Always put the **company's success** ahead of personal accolades.
- Be open and **share knowledge and information** freely.
- Invest time in helping others succeed.
- Be open to the best idea, no matter who it comes from.

6.2.6: Courage

Sometimes, architects have to make tough calls, even if they aren't popular. Having courage is crucial for acting when it truly matters.

- **Speak up** when it's for the greater good, even if it feels uncomfortable.
- Don't shy away from **challenging the status quo** when it's necessary.
- Make **tough decisions** efficiently.
- Take thoughtful risks, even knowing failure is a possibility.
- Stand firm for your **core values**, even when under pressure.
- Be willing to **be vulnerable** to pursue truth and clarity.

6.2.7: Integrity

Trust is everything. Your influence as an architect hinges on your credibility, transparency, and ethical consistency.

- Be known for your **honesty**, **humility**, **and authenticity**.
- Say only what you'd feel comfortable saying **face-to-face** with someone.
- Own up to your mistakes openly.
- Treat everyone with **respect and fairness**, no matter their status or opinion.

6.2.8: Curiosity

Given how fast technology evolves, architects must stay curious to ensure their guidance remains relevant and strategic.

- **Learn quickly and proactively**.
- Look for connections across systems, domains, and ideas.
- Seek out **alternative perspectives**.
- Engage with areas beyond your core specialty to get a broader understanding.

6.2.9: Innovation

While curiosity fuels learning, innovation is what drives impact. Architects should be champions of smart, pragmatic innovation.

- Generate **new ideas** that tackle real problems.
- Strive for elegance by minimizing complexity.
- **Reframe problems** to unlock new solutions.
- Challenge assumptions to uncover better approaches.
- Embrace change, using it as a launchpad for improvement.

6.2.10: Passion

Finally, architects often serve as cultural role models. Passion is what drives excellence, resilience, and inspires those around you.

- **Inspire others** through your commitment and enthusiasm.

This approach not only enhances your journey as an architect but also helps cultivate an environment that values and nurtures leadership in all forms. Let's keep pushing the envelope together!

6.3: Questions to Consider

- *Reflect on the Leader-Leader model of leadership model in your work. How can you empower your team members and encourage them to take ownership of their work?*
- *Have you acted as a facilitator, coach, or mentor as an architect? Can you share an example of when you gave team members guidance, support, or resources to achieve their goals?*
- *How does the phrase "I intend to" resonate with your approach to architecture work? How can it change your perspective on taking the initiative and leading efforts?*
- *How effective do you believe your communication skills are?*
- *How can you foster an inclusive working environment as an architect? How do you nurture and embrace differing perspectives to make better decisions?*
- *Reflect on a situation where you made a decision that was best for the organization rather than what was best for yourself or your group. What was the outcome?*
- *Have you ever had to take an uncomfortable stance but in your organization's best interest?*
- *How do you maintain integrity as a trusted advisor in your organization? Can you share an example where your honesty, authenticity, and transparency were vital?*
- *How have you maintained your curiosity in your role as an architect? Can you share an instance where your learning eagerness led to a significant outcome?*
- *What innovative solutions have you created as an architect? How have these innovations benefitted your organization?*
- *How do you inspire others with your passion for excellence? Can you share an instance where your optimism and tenacity led to a successful outcome?*

7: Architecting Influence: Six Plays for Grounded IT Architects

image by jacoblund from istock

IN THIS SECTION, YOU WILL: Understand how IT Architects can transform their effectiveness and foster a more collaborative, innovative, and grounded practice by consciously applying David Marquet's six linguistic leadership plays to better navigate the complexities of modern technology environments.

KEY POINTS:

- IT Architects can significantly enhance their leadership and influence by adopting six key communication "plays" from David Marquet's "Leadership is Language," moving beyond outdated Industrial Age command-and-control styles.
- These plays—Control the Clock, Collaborate, Commit, Complete, Improve, and Connect—provide practical linguistic tools to foster better thinking (Bluework), more effective execution (Redwork), and stronger team engagement.
- Applying these plays helps architects build trust, flatten power gradients, encourage psychological safety, and unlock discretionary effort, leading to more robust architectural solutions and greater buy-in.
- The principles align directly with a "Grounded Architecture" approach by promoting data-driven decisions, collaborative networks, adaptability, and strategic impact within organizations.
- By consciously changing their language, architects can cultivate a more innovative, resilient, and effective engineering culture.

In today's fast-paced world, architects have evolved beyond just technical designers. They've become **strategic leaders** and **collaborators**, which are essential for tackling the complexities we face and driving real business value. This shift calls for a **sophisticated approach to leadership**—one that moves away from the old "**command and control** " mentality. Those outdated methods, which worked well in the Industrial Age, don't fit in our modern, knowledge-driven environments, especially within IT. Here, architects often need to **lead by influence rather than authority**. So, I'd say communication isn't just a nice-to-have skill; it's the cornerstone of effective leadership.

A great resource that connects with my experiences is L. David Marquet's "**Leadership is Language**." As a former U.S. Navy Captain,

Marquet shows how the **words leaders choose** can dramatically shape their **team culture**, **effectiveness**, and ultimately, success. He makes a compelling case that even **small shifts in language** can lead to **big impacts** on team performance and morale. This view challenges the traditional views of leadership and offers a "**New Playbook**" for the challenges we face today.

At the heart of this new playbook is the understanding that the nature of work has fundamentally shifted. Leadership in the Industrial Age was all about **maximizing efficiency** and ensuring **compliance** with tasks that were often physical and repetitive. In contrast, IT architecture demands a kind of work that is **complex**, **cognitive**, **and collaborative**. This kind of teamwork thrives on **commitment** rather than just **compliance**; it's about unlocking that extra effort and innovation rather than just meeting the bare minimum. I've seen how adhering to outdated language in our leadership practices can hinder creativity and engagement. Organizations that fail to evolve their leadership language within architectural teams may struggle to innovate, adapt, and attract top talent.

The principles outlined in "**Leadership is Language**" resonate with my vision of **"Grounded Architecture." **** Marquet's six leadership "**plays**" give us **practical tools** to enhance communication, encourage deeper thinking, and build more committed teams—all of which help us become more truly "**grounded**." One crucial, yet often unspoken, challenge for IT architects is dealing with the "**power gradient**," or the perceived **hierarchical distance** between people. I've noticed that steep power gradients can stifle creativity, suppress valuable insights, and block open communication, all of which are detrimental to effective architectural practices.

Navigating these dynamics is part of the job, especially in complex organizational structures. If we don't make an effort to address these challenges through thoughtful language and communication, genuine collaboration—one of the key elements of Grounded Architecture—won't thrive. Marquet's strategies can help flatten those gradients and encourage more inclusive and effective interactions, which I believe are vital for our success.

7.1: Setting the Stage: From "Redwork/Bluework" to Effective Action

Before exploring specific plays, it's essential to grasp a fundamental concept from Marquet's framework: the distinction between "Redwork" and "Bluework." This distinction clarifies the different operational modes within any team and highlights the importance of balancing them for optimal performance.

Redwork refers to the "**doing**" **or** "**execution**" phase of any project. It emphasizes performance, **efficiency**, and **minimizing variability** to achieve a specific outcome. In the context of IT architecture, Redwork includes activities such as coding a proof-of-concept based on a defined specification, meticulously documenting a finalized architectural design, or implementing a system according to a detailed plan. The primary focus during Redwork is on execution and **achieving predetermined goals**.

Bluework, on the other hand, represents the "**thinking**" **or** "**decision-making**" phase. This mode **embraces variability** and is characterized by reflection, planning, strategic problem-solving, and collaboration. For IT architects, Bluework is their natural environment; it encompasses strategic design sessions, the analysis of various technological options, the evaluation of emerging technologies, and collaborative workshops aimed at addressing complex architectural challenges.

Marquet stresses the critical importance of **oscillation** between these two modes. Effective teams and their leaders must consciously and deliberately shift between periods of Redwork and Bluework. Being entrenched in one mode while neglecting the other can be detrimental. For instance, excessive Bluework can lead to "analysis paralysis," while an overemphasis on Redwork without sufficient Bluework can result in rushed and poorly conceived solutions.

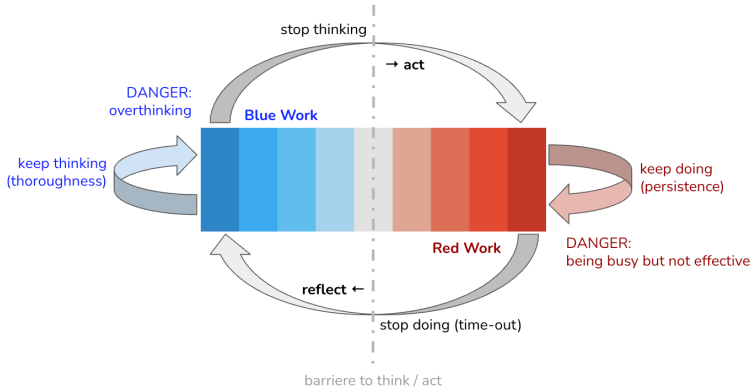


Figure 1: *The dynamic between "Blue Work" (thinking/deciding) and "Red Work" (doing/executing), and the importance of consciously switching between the two modes to lead more effectively. This model encourages a conscious, rhythmic cycle of thinking and doing, where teams pause to reflect, adjust, and act deliberately, breaking the habit of reflexive execution or analysis.*

This Redwork/Bluework framework is profoundly relevant for IT Architects. Their role inherently demands significant time dedicated to Bluework—strategic thinking, careful design, and thorough problem analysis are the bedrock of sound architecture. However, the **relentless pressure for rapid delivery** in many IT environments can inadvertently **push teams, including architects, into a state of perpetual Redwork**, sacrificing crucial thinking time. Recognizing the current "work mode" of the team allows architects to choose the appropriate language and leadership play to either facilitate deeper thinking or drive effective execution.

Failing to distinguish and manage the balance between Redwork and Bluework often leads to common architectural pitfalls. Symptoms like "**analysis paralysis**," where teams become stuck in endless deliberation (excessive Bluework), or "**rushed, flawed implementations**," which stem from inadequate thinking and premature execution (insufficient Bluework before Redwork), are frequent occurrences. Marquet's plays, such as "Control the Clock" to shift into Bluework or

"Commit" to transition into Redwork, offer the necessary mechanisms for effectively navigating these modes.

Moreover, Marquet's framework suggests a **democratization of Bluework**; it should not be the exclusive domain of architects or senior leadership. This perspective challenges traditional hierarchies that may portray architects as the sole "thinkers." For the "Grounded Architecture" approach, which advocates for collaborative networks, architects should use language that actively invites all team members into Bluework activities, such as design sessions or problem-solving workshops. Such **inclusivity enriches the decision-making process** by incorporating diverse perspectives and fostering a shared sense of ownership.

7.2: The Six Leadership Plays in the Architect's Arena

Marquet outlines six specific "plays" that leaders can use to transform their communication and, consequently, their team's performance. These plays offer a new language for leadership, moving away from outdated Industrial Age scripts. Figure 1 illustrates a cyclical framework depicting how IT Architects can enhance their effectiveness and foster a collaborative, innovative, and grounded practice through the intentional application of David Marquet's six linguistic leadership plays:

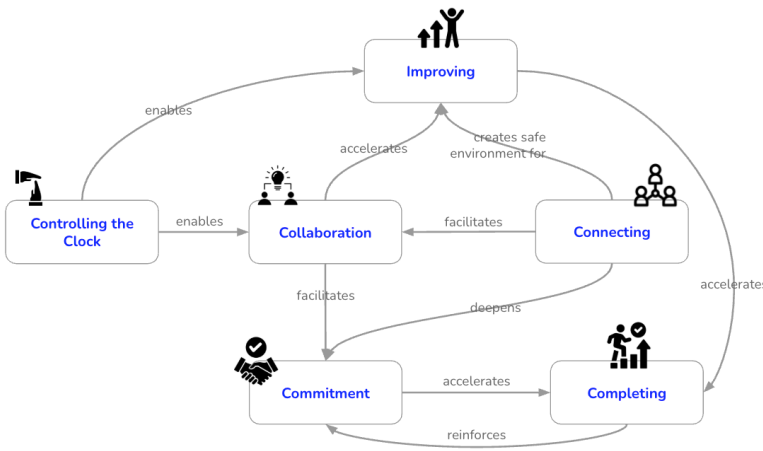


***Figure 2**: David Marquet's six linguistic leadership plays and their key interdependencies.*

The following table provides a concise overview of these plays and their relevance for IT Architects:

Play	Shift From	Key Benefit for IT Architects	Key Marquet Phrase/-Concept
Control the Clock	~~Obeying the clock~~	Enables strategic pauses, better decisions, reduces errors	"Make a pause possible," "Shift to Bluework"
Collaborate	~~Coercing~~	Leverages collective wisdom, fosters innovation, builds buy-in	"Let the doers be the deciders," "Vote first, then discuss"
Commit	~~Complying~~	Drives ownership, unlocks discretionary effort, ensures follow-through	"Commit to learn, not just do," "Discretionary effort"
Complete	~~Continuing indefinitely~~	Provides closure, facilitates learning, focuses on outcomes	"Chunk it small," "Celebrate success"
Improve	~~Proving ability~~	Cultivates a learning culture, continuous enhancement, reduces fear	"What can we learn?" "How can we make it better?"

Play	Shift From	Key Benefit for IT Architects	Key Marquet Phrase/-Concept
Connect	~~Conforming to roles~~	Builds trust, encourages psychological safety, authentic engagement	"Flatten power gradients," "Trust first"

7.2.1: Play 1: Control the Clock, Don't Obey the Clock

The "Control the Clock" play focuses on the intentional act of **pausing action** (Redwork) to allow space for **thinking**, **reflection**, **and decision-making** (Bluework). It challenges the pervasive pressure to **constantly "do"** and promotes **strategic thinking**. Marquet emphasizes that **leaders have a responsibility** not only to make these **pauses possible** but also to **call for them**, especially when teams are deeply immersed in Redwork and may not recognize the need to pause. For instance, language such as, "**We have time to do this right**, **not twice**," can signal that a pause is not only acceptable but encouraged.

image by peopleimages from istock

IT architects often work under **significant pressure to deliver solutions quickly**. The "Control the Clock" play empowers them to carve out crucial moments for strategic architectural reviews. This ensures **alignment with broader goals** and helps **prevent costly errors** that may arise from rushed decisions. This play is instrumental in helping architects avoid the common trap of "**solutioneering**," which is **jumping to solutions before fully understanding the problem** or exploring alternative approaches — a frequent occurrence in fast-paced IT environments.

Examples:

- **Scenario 1 (Design Phase)**: An architect leading the design of a critical new system notices the team converging prematurely on a specific technology choice. They can "control the clock" by saying, "Let's pause the 'how' for a moment. Are we all aligned on the 'what' and 'why'? What are the 2-3 core problems this specific component needs to solve?" This language shifts the team from Redwork (selecting a tool) back into Bluework (clarifying requirements and problem definition).

- **Scenario 2 (Incident Response)**: During a significant system outage, instead of merely issuing directives, an architect might call for a brief "Bluework huddle": "Okay team, let's pause the immediate fixes for ten minutes. What do we know for sure? What are our top two hypotheses for the root cause? What is the safest next diagnostic step we can take?" This practice of calling a time-out, even when not explicitly demanded by the situation, helps normalize the act of pausing.
- **Scenario 3 (Agile Context)**: An architect can proactively incorporate "architectural reflection" slots into sprint planning or review meetings. This ensures dedicated time for Bluework regarding upcoming epics or addressing accumulated technical debt. For example: "Before we commit to these user stories for the next sprint, let's spend 30 minutes discussing the architectural implications of Feature X and any potential long-term impacts."

Effectively "Controlling the Clock" serves as a **crucial prerequisite** for genuine "**Collaboration**" (Play 2) and meaningful "**Improvement**" (Play 5). Without the **intentional pause** created by this play, there is simply no space for **diverse opinions**, **thorough discussions**, or **valuable learning** to take place. If architects do not consciously create these moments for reflection, discussions are likely to be rushed, **dominant voices may overshadow others**, and true collaboration will remain elusive. **Improvement requires reflection**, which is only possible during such pauses. Thus, architects who master this play unlock the potential of several other vital leadership practices.

For architects, this play is also a key mechanism for ensuring their work remains grounded in **strategic objectives** and **data-driven insights**, rather than being swept away by the tide of **short-term project momentum**. **Rushing (obeying the clock)** often leads to **cutting corners** on essential data gathering or strategic thinking. By "controlling the clock," architects create opportunities to **review data**, **consult stakeholders**, and ensure architectural choices **align with broader organizational goals**. This practice helps prevent the "**Ivory Tower" architect syndrome**, where architects become disconnected from practical realities and the needs of the organization.

7.2.2: Play 2: Collaborate, Don't Coerce

The "Collaborate, Don't Coerce" play promotes **genuine collaboration** by actively **inviting and valuing diverse perspectives** rather than allowing leaders to **push their own agendas**—whether subtly or overtly. A central tenet of this approach is to "**let the doers be the deciders**," empowering those **closest to the work** to make meaningful contributions to decisions. Key techniques to foster this collaboration include "**vote first, then discuss**," which helps prevent the group from **anchoring on the leader's opinion**. Leaders are encouraged to **speak last**, after listening to others, and to **cultivate genuine curiosity** about dissenting viewpoints by asking questions like, "What do you see that we don't?" Creating an environment of **psychological safety**, where individuals feel comfortable sharing their honest thoughts without fear of judgment, is essential for the success of this play.

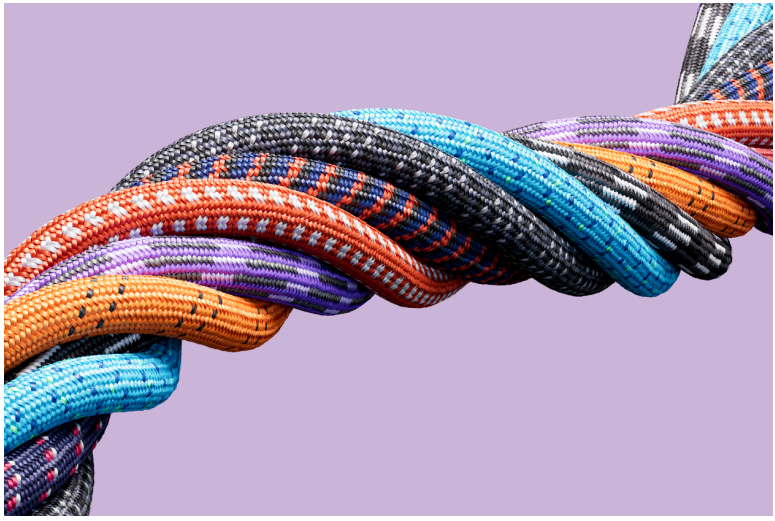


image by olivier le moal from istock

Architects need **buy-in and active participation** from a **diverse array of stakeholders**, including developers, product managers, operations teams, and business units. While coercion might yield superficial

compliance, **genuine collaboration fosters deeper commitment**, more robust solutions, and **shared ownership**. Complex architectural problems, which are common in modern IT, greatly benefit from **cognitive diversity**; collaboration is key to harnessing this collective intelligence.

Examples:

- **Scenario 1 (Technology Selection)**: An architect is tasked with selecting a new messaging queue technology for the organization. Instead of stating their initial preference, they present the problem, outline the selection criteria, and then ask team members to independently write down their top one or two choices along with their reasoning ("vote first"). Following this, the architect facilitates a discussion, ensuring all voices are heard by asking clarifying questions and exploring different perspectives. They provide their own assessment only after everyone else has had a chance to speak.
- **Scenario 2 (Design Review)**: During a review of a proposed microservice architecture, a junior engineer expresses a concern about potential data consistency issues. Instead of dismissing this concern or providing an immediate rebuttal, the architect responds with curiosity: "That's an interesting point. Tell me more about that. What specific scenarios are you envisioning where consistency might become an issue? What do you see that the rest of us might be missing?" This approach validates the contribution and encourages deeper exploration.
- **Scenario 3 (Architectural Principles Co-creation)**: An architect facilitates a workshop with lead engineers from various teams to define a new set of architectural principles for the company. They provide a guiding framework and some initial examples but actively encourage the team to generate, debate, and refine the principles themselves. The architect acts as a facilitator and guide rather than a dictator, asking questions like, "Our goal is to co-create these guiding statements. What are the most critical principles we need to ensure consistency, scalability, and maintainability for our platform moving forward?"

The ongoing trend toward **distributed systems**, and **cross-functional agile teams** in IT makes the "Collaborate, Don't Coerce" play increasingly vital. Architects operating in such environments cannot effectively **dictate solutions from above**; their success relies on their ability to **facilitate**, **integrate**, and **harmonize diverse technical expertise and perspectives**. An architect cannot be the foremost expert in every component of a complex, distributed system, so their capacity to **collaborate effectively with specialists**—rather than **coercing them into a singular**, **preconceived vision**—becomes crucial for architectural quality and adoption. This play is a direct enabler of leveraging the "collaborative networks" central to the Grounded Architecture philosophy.

However, a common pitfall for architects is **"disguised coercion"**—believing they are collaborating when, in fact, they are **subtly steering conversations** toward their preferred outcome. Architects, often being senior and highly experienced, might **unintentionally coerce** through the **strength of their opinions**, the **way they frame questions**, or by **selectively amplifying certain viewpoints**. This play requires **genuine humility** and a **sincere willingness to be proven wrong** or to adopt a solution that differs from one's initial inclination. For some architects, this represents a significant mindset shift, moving away from traditional top-down approaches.

7.2.3: Play 3: Commit, Don't Comply

The "Commit, Don't Comply" play emphasizes obtaining **genuine**, **internally motivated commitment** from the team, which is essential for **unlocking discretionary effort**. This approach contrasts with **mere compliance**, which usually results in only the **minimum effort** necessary to meet requirements. **Commitment is an active choice** made by individuals and is often nurtured through **true collaboration** during the decision-making process. This play encourages **commitment to learning** (rather than executing tasks blindly) and **commitment to actions** (even when individual beliefs or preferences do not fully align with the chosen path). The linguistic shift from *"I can't"* (implying an external force and fostering compliance) to *"I*

don't" (indicating internal resolve and commitment) illustrates this principle.

image by jacob wackerhausen from istock

Architectural standards, **patterns**, and **strategic decisions** are only effective if development teams **genuinely commit** to their adoption and implementation. **Compliance** often leads to **superficial adherence**, **workarounds**, or even eventual **abandonment**. Significant architectural changes, such as **platform modernization** or the **adoption of new paradigms**, require **deep and sustained commitment** from engineering teams to navigate inevitable challenges and successfully complete the initiative.

Examples:

- **Scenario 1 (Adopting a New Standard)**: After a collaborative process (as described in Play 2) to select a new API security standard, the architect seeks explicit commitment from the involved teams. They might say, "We've thoroughly discussed our options and collectively chosen this standard. What support

do you need from the architecture team and from each other to fully commit to implementing this standard for all new services moving forward? What potential roadblocks can we anticipate now and plan for together?" This language positions the adoption as a shared goal and responsibility.

- **Scenario 2 (Proof of Concept)**: An architect initiates a Proof of Concept (PoC) for a new, potentially transformative technology. Instead of merely assigning tasks, they frame the initiative to encourage a commitment to learning: "Our primary goal for this PoC is to *learn* whether this technology can effectively solve X problem for us and to gain a clear understanding of its operational complexities and integration challenges. Let's commit to these specific learning objectives and the actions required to achieve them over the next two weeks."
- **Scenario 3 (Decision Disagreement)**: A team has decided on an architectural approach that the architect has some reservations about, though it's not a critically flawed decision. The architect might express their support by saying, "While I see some potential challenges with this path, the team has made a strong case and collectively decided to proceed. I commit to supporting your decision and will help you succeed in its implementation. Let's agree on the key actions, milestones, and check-in points to ensure we can address any issues that arise." This shows commitment to the team's chosen action, even if the architect's personal belief isn't absolute.

True **commitment** often results from effective **collaboration**. Attempts by architects to secure commitment without first engaging in **genuine**, **inclusive collaboration** are likely to yield, at best, **superficial compliance**. If architects **skip or poorly execute the Collaborate play**—for instance, by **subtly coercing the team toward a predetermined solution**—team members will not feel a **sense of ownership** over the decisions made. Therefore, their adherence will be driven by compliance (*"I'm doing this because the architect said so"*) rather than true commitment (*"I'm doing this because I believe in its value, understand its rationale, and had a meaningful say in the decision"*). This significantly impacts the **quality**, **sustainability**, and **ultimate success** of architectural implementations.

The "Commit, Don't Comply" play is also vital for the effective functioning of **adaptable governance models** (*nudging, taxation, mandates*) as described within the **Grounded Architecture** framework. While **mandates** represent a **top-down enforcement of compliance**, mechanisms like **nudging** and **taxation** rely more on **influencing behavior** and **fostering commitment** rather than imposing strict adherence. These softer forms of governance aim to **guide choices** by making desirable architectural behaviors easier or more attractive, requiring a degree of **voluntary buy-in** or **commitment** from the teams.

7.2.4: Play 4: Complete, Don't Continue

The "Complete, Don't Continue" play challenges the Industrial Age mindset of continuous, undifferentiated, and often unending work. It emphasizes **breaking down tasks** into **defined**, **manageable chunks** with **clear completion points**. A key aspect of this play is the importance of **celebrating successes** upon completion and **learning from each finished cycle**. The principle of *"Chunk it small, but do it all"* is particularly relevant when dealing with **high levels of uncertainty or complexity**. This principle involves **explicitly deciding when to stop** a particular phase of work and having agreed-upon stopping criteria or **definitions of "done"**.

image by photo_concepts from istock

Architectural initiatives—such as **platform migrations**, **system redesigns**, or the **rollout of new enterprise-wide standards**—can be **large**, **complex**, **and long-running**. Breaking these substantial undertakings into **smaller**, **completable phases or milestones** is crucial for **maintaining momentum**, providing opportunities for **iterative learning and adjustment**, and **fostering a sense of accomplishment** within the team. This play helps architects avoid *"architectural drift,"* where designs are endlessly refined without delivering tangible value, or *"analysis paralysis,"* which can **stall progress indefinitely**.

Examples:

- **Scenario 1 (Platform Modernization)**: An architect leading a multi-year platform modernization initiative works with teams to define clear, completable stages with measurable outcomes. For instance: "Stage 1: Migrate User Authentication Service to the new microservices platform. Target completion: End of Q2. Success criteria: 100% of user authentication requests handled by the new service with improved performance metrics." Upon successful completion, the team celebrates this milestone.

- **Scenario 2 (Design Spike)**: When faced with a complex new feature requiring significant architectural design, an architect initiates a time-boxed design spike (e.g., one week) with a specific, completable goal: "By the end of this week, we will have a documented decision on the data storage strategy for Feature Y, including a comparative analysis of the top two alternatives and the rationale for our choice. This document will represent 'complete' for this design spike."
- **Scenario 3 (Architectural Debt Reduction)**: Instead of pursuing a vague and potentially demoralizing goal like "reduce overall technical debt," an architect collaborates with engineering teams to identify specific, completable pieces of technical debt to address each quarter. The successful refactoring or elimination of these items is then acknowledged and celebrated. For example: "This quarter, our focus is to complete the refactoring of the Legacy Billing Module's outdated interface, thereby eliminating a significant source of maintenance overhead."

The "Complete" play creates **defined endpoints** for work increments and incorporates **celebrations of achievement**, directly boosting **team morale** and **reinforcing positive behaviors**. This, in turn, creates a **virtuous cycle**, making future **commitment** (Play 3) easier to achieve. When **successes are recognized and celebrated**, it reinforces the **specific behaviors** that led to those successes, allowing individuals to feel a **tangible sense of accomplishment**. This **positive momentum** and the **trust it builds** make teams more likely to **commit enthusiastically** to the next defined chunk of work.

Furthermore, this play is vital for **demonstrating tangible progress** and **showcasing the value delivered** by the architecture function. This aligns directly with the Grounded Architecture principle, which emphasizes the need to **show impact** and **execute effectively at scale**. Architecture can sometimes be perceived as an **abstract, slow-moving discipline**, leading to accusations of architects operating in an *"Ivory Tower,"* disconnected from **practical delivery**. The **Complete** play focuses on delivering defined architectural outcomes in **manageable chunks**, making the **value of architecture visible, measurable, and timely**. This helps justify architectural investments,

builds credibility for the architecture practice within the organization. Additionally, it provides **clear points** at which *"Lightweight Architectural Analytics"* can be applied to **measure progress**, **impact**, and **adherence to architectural goals**.

7.2.5: Play 5: Improve, Don't Prove

The "Improve, Don't Prove" play advocates for a fundamental shift in mindset—moving away from an environment where individuals feel the need to **constantly prove their competence** and the **infallibility of their decisions**. Instead, it fosters a collective focus on **improving outcomes**, **processes**, and **shared learning**. This play promotes a **culture of curiosity and reflection**, encouraging questions such as *"How could we make this better?"* or *"What can we learn from this experience?"* It develops a **learning culture** where mistakes and setbacks are seen not as reasons for blame or shame, but as **valuable opportunities for growth and refinement**. Leaders embody this mindset by **openly reflecting** on their actions and considering how they could have been improved.

image by sergei chuyko from istock

Architecture is inherently an iterative and evolutionary process; initial designs are rarely perfect and must **adapt to new information**, **changing requirements**, and **feedback from implementation**. An improve mindset allows **architectural solutions to evolve and adapt effectively**. This play is also crucial for **fostering psychological safety** within teams, encouraging engineers and other stakeholders to **identify potential issues or weaknesses** in architectural designs or decisions without fear of retribution or of appearing incompetent.

Examples:

- **Scenario 1 (Post-Incident Review)**: After a significant system failure traced back to an unforeseen architectural flaw, the architect leads a blameless post-mortem. The focus is entirely on learning and improvement: "What can we *learn* from this incident as a team? How can we *improve* our design process, review mechanisms, or monitoring capabilities to prevent similar issues in the future?"
- **Scenario 2 (New Technology Adoption)**: An architect leads the introduction of a new software framework in the organization. Rather than demanding immediate perfection or creating

pressure to prove the framework's viability, they frame the initiative as an opportunity for improvement: "Let's pilot this new framework on Project X. Our main goal is to *improve* our development velocity and code quality for this type of service. What metrics will help us understand if we're achieving that improvement? What challenges are we facing, and how can we adapt our approach or the framework's configuration to overcome them?"

- **Scenario 3 (Architect Self-Reflection)**: An architect shares a learning moment with their team, modeling vulnerability and an "improve" mindset: "Looking back at the initial design for System Z, I realize I didn't fully account for the long-term scalability needs under peak load conditions. That was an oversight on my part. How can *we*, as a team, build in more robust scalability checks and forecasting earlier in our design process next time to avoid this?"

The "Improve, Don't Prove" mindset is fundamental to creating and sustaining a **culture of psychological safety** within technical teams. Without this safety, team members are more likely to **hide mistakes**, **downplay concerns**, or **avoid flagging potential architectural flaws**—all to avoid the risk of **appearing incompetent** or **being blamed** for problems. When the environment prioritizes **proving**, individuals naturally become **risk-averse and defensive**, which **stifles learning and innovation**.

Architects who **champion the improve mindset** actively **cultivate the safety** needed for **open communication** about what is not working. This dialogue is critical for **identifying and addressing architectural weaknesses** before they escalate into major issues. As a **significant cultural enabler**, this play supports the core themes of **adaptability** and **continuous learning** central to the Grounded Architecture philosophy. Grounded Architecture is designed for fast-moving global organizations and emphasizes the need for adaptability in architectural practices and solutions.

The "Improve, Don't Prove" play **institutionalizes the process of learning and adaptation**. By consistently asking, *"How can we*

make it better?", architects ensure that both the architecture and the surrounding practices are **continuously refined and enhanced**. This approach aligns perfectly with the need for an **adaptable and evolving architectural strategy** capable of responding to **changing business needs and technological landscapes**.

7.2.6: Play 6: Connect, Don't Conform

The "Connect, Don't Conform" play encourages leaders to **transcend the limitations of hierarchical roles** and the **implicit expectations of conformity**. Instead, it emphasizes the importance of **building genuine human connections** with team members. This approach involves **intentionally flattening power gradients**, valuing individuals for their **unique perspectives and contributions**, and fostering an environment of **mutual respect and trust**. Leaders are encouraged to **demonstrate vulnerability**, **show care** for their team members' thoughts, feelings, and personal goals, and **extend trust first** rather than waiting for it to be earned. The play contrasts the Industrial Age tendency to conform to hierarchical positions with the **modern need to connect** with others as individuals.

image by pakin jarerndee from istock

IT architects often find themselves in roles where they need to influence outcomes **without having direct managerial authority** over all stakeholders. In these situations, **building trust and rapport** through **genuine human connection** is far more effective than relying on **job titles** or **perceived hierarchical standing**. Additionally, **understanding the diverse perspectives, motivations, and concerns** of various teams—such as development, operations, security, and product management—is crucial for creating **holistic**, **well-rounded**, and **widely accepted** architectural solutions. This deep understanding is cultivated through connection, not conformity.

Examples:

- **Scenario 1 (Cross-Team Collaboration)**: An architect working on a complex, enterprise-wide initiative makes a deliberate effort to have informal conversations (e.g., virtual coffee meetings or brief one-on-one chats) with key members of different engineering teams. The purpose is not solely to discuss the project agenda but to better understand their experiences: "I'd like to understand your team's perspective more deeply. What

are your biggest pain points with the current platform, and what are your aspirations for how it could better support your work?" This indicates a care for people's thoughts and feelings.

- **Scenario 2 (Handling Dissent)**: In a design meeting where a particular architectural proposal is being debated, one engineer persistently voices strong criticisms. Instead of shutting down the critique or becoming defensive, the architect employing the "Connect" play might say: "I appreciate you pushing back on this and clearly sharing your concerns. It's evident you have strong reservations. Let's ensure we fully understand your viewpoint. Help me see what you're perceiving." This language values the individual and their input, fostering a more open dialogue beyond simple role-based interaction.
- **Scenario 3 (Architect Admitting Uncertainty)**: When faced with a novel and particularly challenging technical problem for which they don't have an immediate solution, an architect shares their uncertainty with the team: "Honestly, I don't have a ready-made answer for this one. This is new territory for me as well. Let's explore this challenge together. What are your initial thoughts, ideas, or even gut feelings on how we might approach this?" Admitting "I don't know" demonstrates vulnerability, builds trust, and invites collaborative problem-solving.

"Connect, Don't Conform" serves as an **overarching strategy** that enhances the effectiveness of all other leadership plays. Without **genuine connection**, **mutual respect**, and **trust**, attempts to truly *Collaborate*, gain deep *Commitment*, or foster a safe environment to *Improve* are likely to be **less effective and more superficial**. If team members feel they are merely expected to conform to their roles or to the perceived hierarchy—acting as **cogs in a machine**—they are less likely to **openly share their best ideas**, **genuinely commit to decisions**, and will certainly not feel **safe enough to point out flaws or suggest improvements**.

Connection builds the **psychological safety and trust** that are essential for **positive team interactions**. It forms the **relational foundation** upon which effective architectural leadership is built. This play is crucial for nurturing **collaborative networks** and fostering

an understanding of **human factors**, which are central pillars of the Grounded Architecture philosophy. Grounded Architecture emphasizes the importance of *Collaborative Networks* and dedicates attention to *On Human Complexity*, which includes understanding cultural differences and cognitive biases. The act of **connecting** is precisely how architects build these vital relationships.

7.3: IV. Conclusion: Integrating Leadership Principles into Your Architectural Practice

The six leadership principles outlined in David Marquet's *Leadership is Language—Control the Clock*, *Collaborate*, *Commit*, *Complete*, *Improve*, and *Connect*—provide a **valuable toolkit for IT Architects** looking to enhance their **leadership effectiveness**. By adopting these linguistic shifts, architects can transform their leadership style from a traditional, potentially less impactful approach to one that is more **empowering**, **collaborative**, and **results-oriented**. These concepts are not just abstract theories; they are **practical**, **actionable tools** that can be implemented immediately through a **conscious change in the language** we use in our daily interactions.

Architects should start by **observing their own language** and the **prevailing communication patterns** within their teams and organizations. Which principles resonate most strongly with them? Which ones address the current challenges they face? **Starting small**, perhaps by focusing on intentionally applying one or two principles, can help **build confidence** and **demonstrate tangible benefits**. **Consistent application** of these principles by architects can create a **positive ripple effect**, influencing not only their **immediate teams** but also promoting a **more constructive and innovative engineering culture** across the broader organization. As architects model this new language of leadership, others may begin to adopt similar communication patterns. This can lead to a **gradual but significant positive shift** in how technical discussions are conducted, **decisions are made**, and how **individuals and teams interact**, ultimately fostering a **more psychologically safe and innovative environment**.

Embracing these leadership principles aligns directly with the core tenets of Grounded Architecture. By mastering this new language, architects can **build stronger and more effective collaborative networks**, make **more robust and data-informed decisions** by creating the necessary space for **diverse input** and thorough analysis, enhance the **adaptability** of their architectures and practices, and

deliver greater strategic value to their organizations. The journey to becoming a truly *Grounded Architect* is, in many ways, a journey of mastering the language of modern leadership. This transformation is not a one-time event but a **continuous path of improvement**, similar to mastering any complex technology or methodology. It requires ongoing practice, conscious reflection, and a **willingness to adapt and refine** one's approach over time. This **commitment to evolving one's language** is a hallmark of an architect dedicated to not only technical excellence but also to **impactful and empowering leadership**.

7.4: Questions to Consider

- *Which of the six leadership plays (Control the Clock, Collaborate, Commit, Complete, Improve, Connect) do you feel is most lacking in your current team interactions, and what's one small linguistic change you could make this week to start addressing it?*
- *Think about a recent architectural decision or discussion. How could applying the "Control the Clock" play (e.g., calling a deliberate pause for Bluework) have potentially improved the outcome or the process?*
- *When was the last time you consciously used language to "Collaborate, Don't Coerce"? How did you ensure diverse perspectives were heard before your own (e.g., "vote first, then discuss")?*
- *Consider a current architectural standard or initiative. Are your teams truly "Committed," or are they merely "Complying"? What language could you use to shift towards genuine commitment?*
- *How can you apply the "Complete, Don't Continue" play to break down a large, ongoing architectural effort into smaller, more manageable, and celebratable milestones?*
- *Reflect on a recent project setback or architectural challenge. How could an "Improve, Don't Prove" mindset have changed the team's approach to learning from the experience?*
- *In what ways can you intentionally "Connect, Don't Conform" to build stronger relationships and flatten power gradients with stakeholders outside your direct team?*
- *How often do you find yourself or your team stuck in "Redwork" (doing) without sufficient "Bluework" (thinking/planning)? What triggers could you establish to prompt a shift?*
- *How might the "power gradient" in your organization be subtly influencing architectural discussions, and what specific phrases from Marquet's plays could help mitigate this?*
- *What is one "Industrial Age" leadership phrase you commonly use or hear that you could replace with a "New Playbook" alternative to foster better architectural outcomes?*

8: Balancing Curiosity, Doubt, Vision, and Skepticism in IT Architecture

image by vawiley from istock

IN THIS SECTION, YOU WILL: Understand that balancing curiosity, doubt, vision, and skepticism is essential for driving sustainable innovation and change in organizations.

KEY POINTS:

- **Curiosity and wonder** spark exploration, leading to technological breakthroughs. However, without caution, it can result in premature adoption of immature solutions.
- **Doubt** forces a critical evaluation of progress, ensuring that innovative ideas are grounded in practical and validated approaches. Over-reliance on doubt can stifle risk-taking and hinder breakthrough innovation.
- **Vision and belief** provide a long-term perspective, guiding efforts toward significant, transformative goals. Yet unchecked vision can lead to confirmation bias or misdirected resources.
- **Skepticism** helps prevent overcommitment to unproven ideas, ensuring teams remain grounded. However, excessive skepticism can result in missed opportunities and discourage creative risk-taking.

- Architects must constantly reflect on how well they balance these forces. In doing so, they can ensure that their efforts are driven by a healthy combination of exploration, critical validation, strategic guidance, and cautious realism—all crucial to achieving lasting success in any organizational change.

In today's fast-paced tech world, **IT architects are crucial** in helping organizations innovate and transform in ways that are both **responsible and sustainable**. They're not just bringing fresh ideas to the table; they also evaluate and guide the ideas of others, striking a balance between seizing opportunities and maintaining oversight.

To foster **sustainable innovation** and navigate **effective organizational change**, architects need to understand the **motivational forces** that drive individuals and teams. Drawing from my previous research, I like to think of these forces as a **compass** that points to four key motivators essential for guiding responsible innovation:

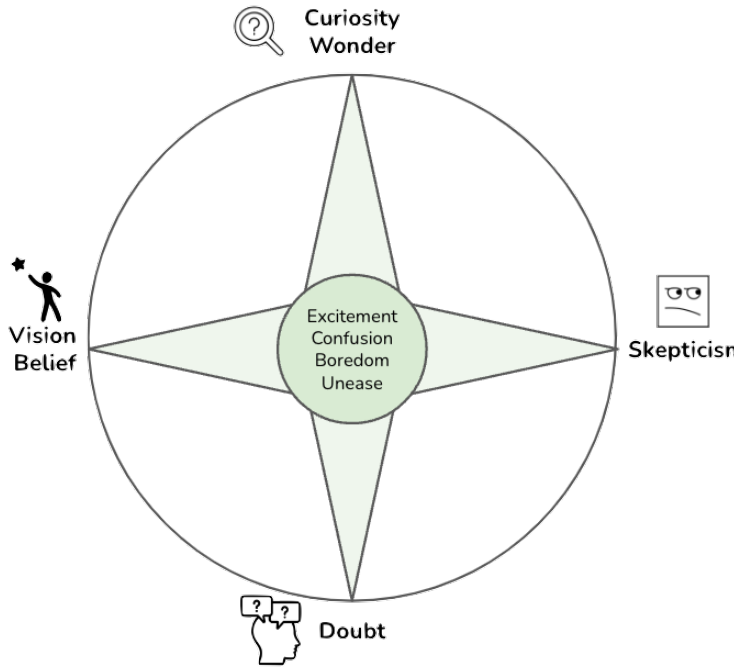


***Figure 1**: The compass for driving sustainable innovation and change in organizations.*

Each direction on this compass stands for a different but interconnected motivator:

- **Curiosity** encourages exploration and the hunt for new possibilities.
- **Doubt** prompts us to reflect critically and reevaluate our assumptions.
- **Vision** provides clarity by linking our efforts to long-term strategic goals.
- **Skepticism** ensures we challenge feasibility and practicality, paving the way for sound execution.

These four forces—**explorative**, **reflective**, **strategic**, **and critical**—sometimes pull us in different directions. However, for innovation to

truly be sustainable, the objective isn't to choose one over the others. Instead, it's about **keeping a thoughtful balance** among them.

Architects are in a unique position to recognize and weave these four motivators into their innovation and transformation initiatives. They play a key role in making sure that:

- **Curiosity** sparks structured experimentation rather than a chaotic free-for-all.
- **Doubt** and **skepticism** sharpen our thinking without putting a damper on our creativity.
- **Vision** aligns innovation with the organization's broader goals and impact.

By helping teams find this balance, architects **enable innovation that is both bold and grounded**, **strategic yet adaptable**. This delicate equilibrium empowers organizations to pursue change with both **imagination and discipline**, a combination that's essential for long-term success.

In the upcoming sections, I'll dive deeper into each of these four forces, showcasing how they influence innovation and how architects can effectively activate and harmonize them within their teams and across the organization. Let's explore this journey together!

8.1: Curiosity and Wonder: The Spark of Innovation

When we think about what drives innovation, **curiosity and wonder** often stand out as some of the most powerful motivators. Curiosity is that innate desire to explore, understand, and ask, *"What if?"*—and it's a crucial force behind technological progress.

In vibrant engineering cultures, curiosity isn't just welcomed; it's nurtured. When organizations foster **autonomy**, **exploratory freedom**, and **space for experimentation**, they create fertile ground for curiosity to flourish—and guess what? Innovation follows naturally.

Image of Curiosity in Action

image by mtstock studio from istock

8.1.1: The Architect's Role in Enabling Productive Curiosity

It's interesting to note that architects—who often play a pivotal role in guiding innovation—can sometimes lose that spark of curiosity

over time. They might find themselves bogged down by reviewing others' ideas or focusing too much on risk management. But the best architects embrace their role as **curious catalysts**. They leverage their wide-ranging knowledge to explore and assess emerging technologies **with intention and purpose**.

Architects hold a unique position to drive **responsible exploration**—it's about more than just following trends; it's about understanding their implications and potential impacts.

According to insights from ThoughtWorks[1], there's a growing **gap between the pace of technological change and how quickly organizations can adopt these changes**. When organizations don't keep up, they're not necessarily lacking in innovation; instead, they struggle to **absorb and apply** new ideas in a sustainable fashion.

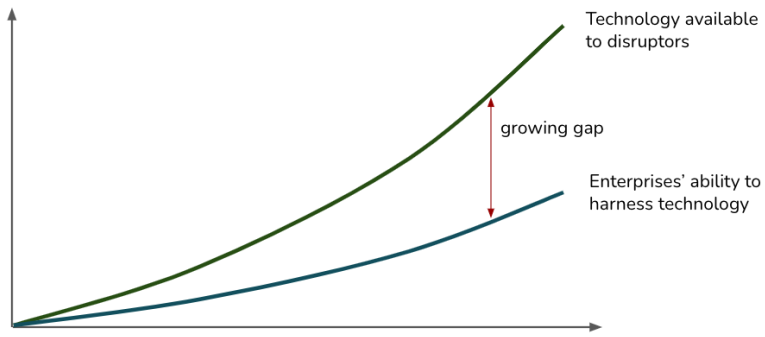


Figure 2: *Technology advances exponentially, but organizations struggle to adopt at the same pace. Curiosity helps bridge this gap. Source: thoughtworks.com*

[1] https://www.thoughtworks.com/en-ec/insights/blog/seven-tenets-new-age-business-analysis

8.1.2: The Double-Edged Nature of Curiosity

Now, while curiosity is a fantastic driver of progress, it can also lead teams astray if it's not kept in check. If we dive headfirst into emerging technologies without a critical eye, we might end up with **naive solutions**, **overcomplicated systems**, or **unnecessary technical debt**.

To strike the right balance, curiosity needs a counterweight—like doubt, skepticism, and a long-term vision—so we don't end up chasing novelty for the sake of novelty.

8.1.3: Real-World Examples: Curiosity in Action (and It Risks)

Here are some real-world examples of how curiosity has spurred meaningful innovation, along with the pitfalls that sometimes come with it:

8.1.3.1: Microservices Architecture

- **Innovation driver**: Curiosity about modularity and scalability gave rise to microservices, offering flexible deployments and cloud-native ecosystems.
- **Sustainability risk**: Teams that rushed into microservices without the right tools or operational readiness faced increased complexity and performance issues.

8.1.3.2: Serverless Computing

- **Innovation driver**: Developers eager to reduce infrastructure burdens pushed the adoption of serverless architectures.
- **Sustainability risk**: Those who didn't consider factors like cold start times or vendor lock-in often ran into performance bottlenecks later.

8.1.3.3: Agile Methodology

- **Innovation driver**: The desire for quicker, more adaptable workflows fueled the rise of Agile.
- **Sustainability risk**: Misusing "Agile in name only" created chaotic environments without actually improving outcomes.

8.1.3.4: Blockchain in Software Development

- **Innovation driver**: The attraction of decentralization and security ignited experimentation in numerous non-financial areas.
- **Sustainability risk**: Occasionally, blockchain was applied where more straightforward technologies would have sufficed, leading to unnecessary complexity.

8.1.4: Final Thought: Channeling Curiosity with Purpose

Curiosity is vital for maintaining relevance and competitiveness—it's what drives learning, experimentation, and innovation. But **curiosity on its own isn't enough**. We need to pair it with **critical thinking, thoughtful evaluation, and strategic alignment**.

Architects have a key role in finding this balance: empowering teams to explore boldly while ensuring their pursuits are **sustainable, purposeful, and in tune with real-world needs**. So let's encourage that spark of curiosity—but let's do it wisely!

8.2: Doubt: The Key to Certainty and Strength in Innovation

You know that feeling you get when you start making real progress on a project? It's exciting! But then suddenly, doubt creeps in. You ask yourself, *"Is this actually right?"* or *"Am I just imagining these results?"*

Well, here's the thing: doubt isn't just a nagging voice in your head—it's one of the most **overlooked yet crucial forces in sustainable innovation**.

Instead of being a negative force that holds us back, doubt actually motivates us to **validate our assumptions**, **refine our ideas**, **and enhance our designs**. It's that inner whisper encouraging us to ask, *"Have we really thought this through?"* And by doing so, it **elevates the quality and reliability of our work**.

image by hiraman from istock

8.2.1: Doubt vs. Skepticism: What's the Difference?

Let's clarify something: doubt and skepticism are not the same. **Skepticism** questions the entire foundation of what we're doing, even whether it's worthwhile at all. In contrast, **doubt** acknowledges that we're on the right track but asks, "How can we make this even better?"

Doubt aims to ground our ideas more firmly, not dismiss them outright.

In fields like architecture and software engineering, doubt manifests through practices that promote quality—think testing, peer reviews, validation frameworks, and architectural scrutiny.

8.2.2: Building a Culture of Constructive Doubt

As IT architects, it's essential to nurture a culture where doubt is seen as an asset, not a hindrance. When embraced constructively, doubt can lead to **stronger systems**, **clearer thinking**, **and smarter decision-making**.

However, there's a catch: **too much doubt can be counterproductive**. If we allow ourselves to get stuck in analysis paralysis, we risk stifling innovation and avoiding meaningful risks. The trick is finding **balance**—using doubt to challenge our assumptions while still encouraging bold thinking.

8.2.3: Real-Life Examples of Doubt in Architecture

8.2.3.1: Unit Testing & Test-Driven Development (TDD)

- **How doubt helps**: Writing tests before coding ensures we validate behavior and minimize future failures—grounding our work in real, testable outcomes.
- **When doubt goes too far**: If tests become too rigid, they can discourage change and make it harder to refactor, locking teams into overly cautious patterns.

8.2.3.2: Code Reviews

- **How doubt helps**: Code reviews bring in fresh perspectives, catch flaws, and boost maintainability. They promote a sense of shared ownership.
- **When doubt goes too far**: If reviews become excessively critical or perfectionistic, they can slow down progress and dampen team morale.

8.2.3.3: CI/CD Pipelines

- **How doubt helps**: Automated validation checks that our software behaves correctly at every stage of integration and deployment.
- **When doubt goes too far**: Teams might hesitate to make big changes or embrace architectural innovations if they fear triggering failure flags in closely monitored pipelines.

8.2.3.4: Architectural Peer Reviews

- **How doubt helps**: Peer reviews help identify blind spots in our system design and ensure we align with non-functional requirements while minimizing long-term risks.
- **When doubt goes too far**: Reviews can sometimes create a risk-averse culture that discourages exploring new patterns or technologies.

8.2.3.5: Security Testing (Penetration Testing & Threat Modeling)

- **How doubt helps**: Security testing operates on the assumption that vulnerabilities exist, driving us to build stronger defenses and resilience.
- **When doubt goes too far**: Being overly cautious can hinder innovation and slow down feature delivery, resulting in overly restrictive designs.

8.2.4: A Final Thought: Doubt as a Design Principle

When used constructively, doubt can transform innovation into resilience. It ensures that our new ideas can withstand scrutiny—and that every spark of enthusiasm is backed by rigor.

Doubt doesn't slow us down; it actually strengthens our progress.

But for that to happen, it needs to be balanced with curiosity, vision, and a readiness to act. For IT architects, this means creating an environment where teams feel free to **ask tough questions**, **validate openly**, **and still advance with confidence**.

8.3: Vision and Belief: Sustaining Transformation Through Purpose

Let's talk about the power of *belief* in driving innovation and transformation. While "belief" sometimes sounds like blind faith, in our world of change, it's absolutely **essential**. Think about it: meaningful change is hard to picture without a guiding vision—a strong conviction in something better on the horizon.

Vision and belief are the long-term motivators that keep us going. They fuel the energy and focus needed for big initiatives, often long before we see any results. Many architectural transformations kick off not with data or doubts, but rather with **bold ideas and a strong belief in what could be**.

image by georgepeters from istock

8.3.1: Vision Is Direction—Not Destination

While **curiosity** encourages us to explore, vision gives us the direction we need. Where **wonder** motivates us to ask questions, belief drives

us to stay purposeful and resilient. Together, they create the engine of innovation: **explore widely, but stay focused**.

Having a vision takes sustained effort, clarity, and the ability to lead others through the unknown. However, it's important to remember that vision without critique can lead to **confirmation bias**, where teams might overlook challenges while trying to validate their own ideas. This is why architects need to **balance visionary thinking with the practical forces of doubt, skepticism, and experimentation**.

8.3.2: Real-World Examples of Vision in Action

Let's look at some examples where **vision and belief sparked technological transformation**, but critical evaluation ensured those visions became reality:

8.3.2.1: The Internet

- **Visionary spark**: Think about visionaries like Vannevar Bush and J.C.R. Licklider, who dreamed of a global interconnected network—what Licklider dubbed an "Intergalactic Computer Network"[2].
- **Balancing belief**: This imaginative vision only became a reality through decades of tackling challenges around protocols, bandwidth, and reliability—all grounded in rigorous engineering and continuous improvement.

8.3.2.2: Agile Software Development

- **Visionary spark**: The Agile Manifesto captured a belief that software development could be quicker, more adaptable, and more centered on human needs.
- **Balancing belief**: Agile success hinges on teams continuously testing and refining their approaches. When belief outpaces reality—like in cases of "Agile in name only"—results can fall short.

[2] https://en.wikipedia.org/wiki/Intergalactic_Computer_Network

8.3.2.3: The DevOps Movement

- **Visionary spark**: DevOps emerged from the belief that breaking down silos can enhance collaboration, speed, and quality in software delivery.
- **Balancing belief**: DevOps thrives when teams take a pragmatic approach—adapting tools, processes, and culture thoughtfully, rather than following a strict dogma.

8.3.2.4: Artificial Intelligence (AI) and Machine Learning (ML)

- **Visionary spark**: From Alan Turing's pioneering ideas to today's advancements in NLP, computer vision, and generative models, belief in AI's potential is pushing the envelope of innovation.
- **Balancing belief**: Responsible AI development demands rigorous testing, ethical considerations, and a grounded understanding of limitations—balancing hype with humility.

8.3.3: Vision Without Balance Is Risky

While vision and belief are crucial, **they alone aren't enough**. Without curiosity, we miss out on exploring alternatives. Without doubt, we risk overlooking our assumptions. And without skepticism, we could be building illusions on shaky ground.

The best architects are the ones who can **inspire belief** while remaining **open to challenge and adaptation**.

8.3.4: Final Thought: The Courage to Believe, the Discipline to Validate

In the realm of architectural leadership, **vision infuses innovation with meaning**. It transforms a scattered collection of ideas into a

united movement, rallying people around shared goals and providing long-term focus in our fast-evolving tech landscape.

But let's keep this in mind: belief must be tested, and vision must be earned—**not just through ideas, but through execution**.

When architects strike a balance between belief and rigor, they don't simply follow trends—they help shape the future.

8.4: Skepticism: The Critical Lens That Fuels Innovation

Let's talk about skepticism. It often gets a bad rap, seen as cynicism or resistance to change. But in the world of IT architecture, skepticism is more like a smart, informed reality check. According to Wikipedia, it's about having "doubt regarding claims that are taken for granted elsewhere." This means skepticism helps organizations steer clear of unexamined assumptions and potential pitfalls.

image by izusek from istock

When doubt prompts us to seek more certainty, skepticism nudges us to pause and reconsider—sometimes even to abandon certain ways of thinking. It raises essential questions: *Is this really feasible? Is this solution based in actual reality?*

8.4.1: Skepticism ≠ Pessimism

Here's the important part: skepticism isn't about dismissing innovation. It's about ensuring that any new ideas are thoughtful, practical, and grounded in the real world. Fred Brooks's classic paper, "No Silver Bullet—Essence and Accidents of Software Engineering,"[3] serves as a great example. He challenged the notion that a single breakthrough could drastically improve software productivity. Instead, he argued for progress through smaller, more feasible innovations.

Skepticism is not pessimism—it is realism tempered with experience.

8.4.2: How Skepticism Shapes IT Architecture

In the realm of IT architecture, skepticism plays a pivotal role in:

- Evaluating new trends critically
- Identifying hidden costs or complexities
- Preventing over-engineering or the premature adoption of unproven technologies

That said, **unchecked skepticism** can also create problems. It can lead to missed opportunities, stifle promising innovations, and breed a culture of stagnation. When skepticism becomes a reflex instead of a reasoned approach, it slides into unproductive negativity.

So, what we need is **practical skepticism**. This goes beyond instinct; it requires **deep knowledge**, **experience**, **and a well-rounded perspective**.

8.4.3: Real-World Examples of Constructive Skepticism

Let's look at some real-life examples where skepticism had a positive impact:

[3] URL_3

8.4.3.1: Monolith vs. Microservices

- **Skeptical insight**: Microservices seem great for flexibility and scalability, but early adopters raised concerns about distributed complexity and communication overhead.
- **Impact**: This skepticism led to hybrid models—like *modular monoliths*—that preserve simplicity while offering scalability. The result? Tailored solutions that truly fit the problem.

8.4.3.2: Blockchain Everywhere

- **Skeptical insight**: Blockchain was touted as a magic solution for everything from supply chains to voting systems. Critics questioned whether its complexity was genuinely necessary.
- **Impact**: This critical viewpoint prevented many organizations from misusing blockchain. Instead, they focused on traditional databases when decentralization wasn't vital.

8.4.3.3: Artificial Intelligence (AI) Hype

- **Skeptical insight**: Skeptics pointed out the exaggerations surrounding general AI and autonomous vehicles, particularly regarding safety and data bias.
- **Impact**: This grounded view fostered ethical AI development, promoting realistic use cases (think narrow AI) and better safeguards for deployment.

8.4.3.4: Agile Methodology Skepticism

- **Skeptical insight**: Agile doesn't work for every team. Architects highlighted concerns where Agile was applied without considering team maturity, product type, or architectural needs.
- **Impact**: This pushback led to innovative **hybrid methodologies**, blending Agile practices with upfront planning, particularly for larger systems.

8.4.3.5: Serverless Computing

- **Skeptical insight**: While serverless platforms promise speed and scalability, skeptics pointed out issues like cold starts and vendor lock-in.
- **Impact**: Many teams adopted **selective serverless strategies**, using them for event-driven workloads while keeping traditional infrastructure for core systems.

8.4.4: Skepticism as a Leadership Skill

Skepticism is arguably one of the most **intellectually demanding** skills for leaders. Unlike doubt or curiosity, which can rely on structured questions, skepticism leans on **judgment**, **pattern recognition**, **and a broad awareness of systems**.

There aren't many structured tools for skepticism. Instead, it comes from **experience**, **deep expertise**, **and a willingness to challenge the status quo**.

The best architectural leaders aren't just contrarians—they're **constructive skeptics**. They know when to question the hype and when to champion tried-and-true solutions.

8.4.5: Final Thought: Finding Balance

In the end, skepticism is more of a safeguard than a roadblock. It invites us to think critically, ensuring that our innovations truly resonate with the realities we face. Balancing healthy skepticism with openness to new ideas can lead to solutions that are not just innovative, but genuinely effective.

8.5: Integrating the Compass: Finding Balance in Sustainable Architecture

Imagine the four points of a compass—**curiosity**, **doubt**, **vision**, **and skepticism**. Rather than being opposing forces, they actually complement each other beautifully. When we strike a balance among these motivators, we lay the groundwork for resilient and forward-thinking IT architecture.

image by happyphoton from istock

In high-performing teams, these motivators work hand in hand:

- **Curiosity** ignites our desire to explore and learn new things.
- **Doubt** encourages us to test and validate our ideas.
- **Vision** keeps us focused on a clear direction and purpose.
- **Skepticism** makes sure we stay grounded in reality.

Now, if we lean too heavily on any one of these forces, we might find ourselves off-balance. For example, too much curiosity can lead

to uncritical innovation, while too much skepticism might result in stagnation. The key? When we use all four motivators together, we make decisions that are **bold yet cautious**, **strategic yet adaptable**, **and ambitious yet realistic**.

8.5.1: Cultivating Balance Through Questions

A great way for IT architects to help their teams integrate these forces is by regularly asking the right questions:

- **Curiosity**:
  - Are we actively exploring new technologies and approaches, or are we just sticking to what we already know?
  - Are we encouraging room for experimentation and learning throughout the organization?
- **Vision**:
  - Do we have a clear architectural direction guiding our efforts?
  - Are we just reacting to trends without a cohesive strategy in place?
- **Doubt**:
  - Are we using the right methods—like testing and validation—to reduce uncertainty and boost our confidence in our solutions?
  - Are we willing to refine our ideas when necessary?
- **Skepticism**:
  - Are we critically examining our own assumptions and biases?
  - Are we being skeptical of our own preferred solutions, or just questioning those from others?

8.5.2: Final Thought: Leading with Balance

IT architects play a crucial role in **orchestrating this balance**. By blending curiosity, doubt, vision, and skepticism, they can create environments where:

- New ideas are explored boldly but assessed wisely,
- Strategy informs action without stifling innovation,
- Critical thinking enhances outcomes, rather than slowing them down.

The aim isn't just to pick a direction but to build a compass that keeps the team moving forward with purpose, clarity, and confidence.

When these four motivators align harmoniously, organizations are not only better equipped to navigate complexities but also to embrace change and drive lasting innovation. Let's embrace this balanced approach and see where it takes us!

8.6: To Probe Further

- The Four Points of the Research Compass[4], by Željko Obrenović, 2013

[4] https://researcher-practitioner.com/research-compass

8.7: Questions to Consider

- *How does curiosity drive innovation in your organization, and how do you prevent it from leading to unsustainable or risky decisions?*
- *In what ways can doubt contribute to the quality and robustness of your work, and how do you ensure that it doesn't stifle innovation?*
- *How do you balance long-term vision and belief with the need for critical evaluation and validation in your decision-making?*
- *How do you cultivate constructive skepticism without allowing it to turn into cynicism that hampers creativity?*
- *When exploring new technologies, how do you ensure you follow industry trends and align them with a clear strategic vision?*
- *How can fostering a culture of curiosity, doubt, vision, and skepticism lead to more resilient and impactful solutions in your organization?*
- *What strategies can you use to balance risk-taking with rigorous validation in the innovation process?*
- *How does your organization support contributions driven by the four motivators: curiosity, doubt, vision, and skepticism?*
- *In what ways can skepticism be used as a tool for improvement without dismissing promising new ideas too early?*
- *How do you foster environments where curiosity and exploration are encouraged but tempered with thoughtful, critical evaluation?*