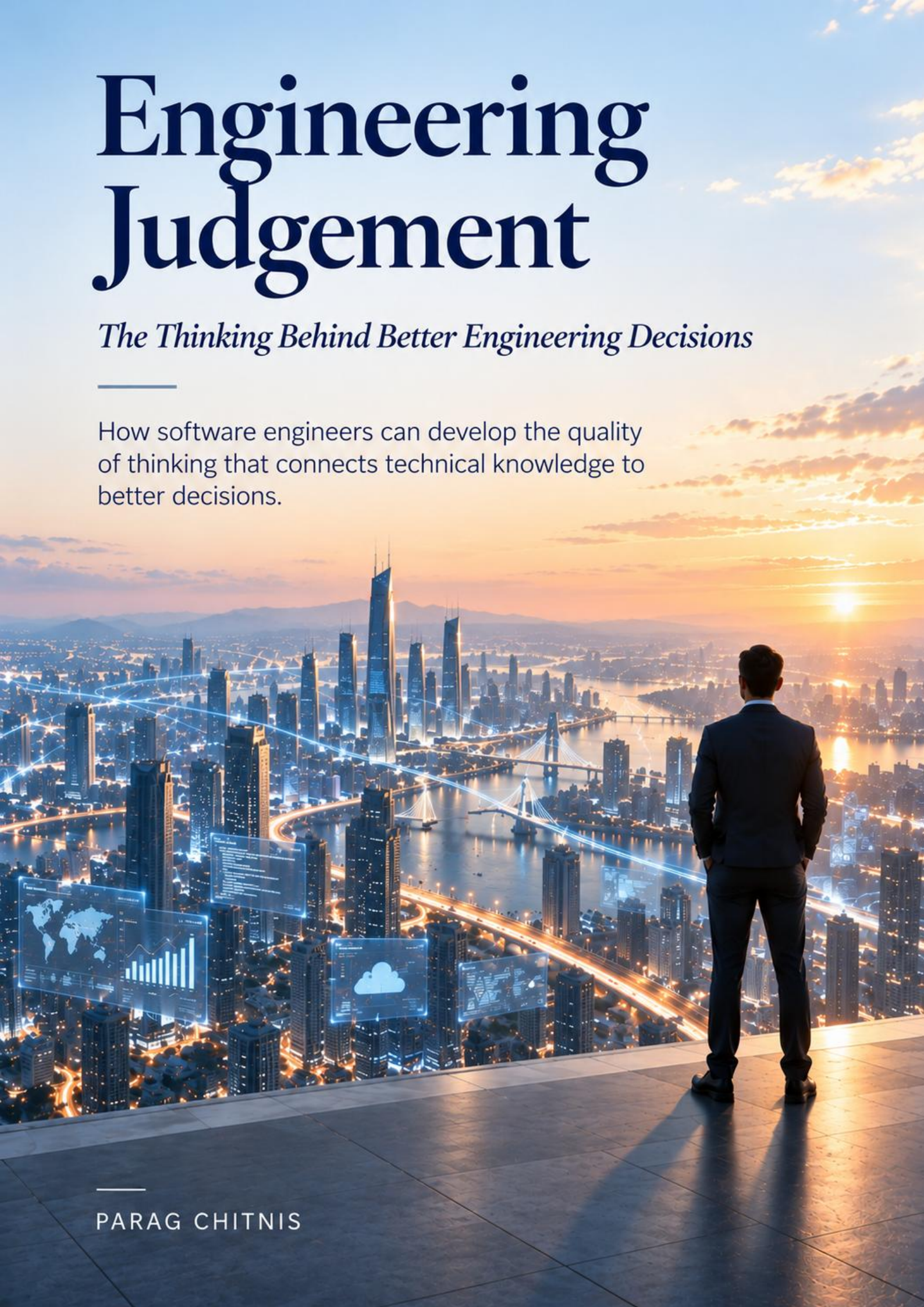
Engineering Judgement

The Thinking Behind Better Engineering Decisions

How software engineers can develop the quality of thinking that connects technical knowledge to better decisions.



PARAG CHITNIS

Engineering Judgement

The Thinking Behind Better Engineering Decisions

This paper explores how software engineers can develop judgement, and why it is one of the defining capabilities of modern software engineering.

Author

Parag Chitnis

Website

paragchitnis.com

Version 1.6

Contents

CHAPTER 1

The Engineers Everyone Trusts

Walk into almost any engineering organization and you'll notice something interesting. There are engineers people consistently turn to when decisions become difficult.

When a production system is failing, they are the ones people look to for clarity. When a team cannot agree on an architectural direction, they are the ones called in. When a customer escalates or a deadline suddenly changes, they are the ones everyone wants in the room.

These engineers aren't always the loudest voices in the room. They are not always the most experienced, nor are they necessarily the deepest technical specialists. Yet people trust them. Not because they always have the answer, but because they seem to make sense of situations that appear confusing to everyone else.

They remain thoughtful when others become reactive. They notice risks before they become problems. They recognize assumptions that everyone else has accepted without question. They anticipate consequences that only become obvious months later. So when they recommend a course of action, people listen.

We Recognize It, But Rarely Name It

Engineering leaders encounter these individuals throughout their careers. They often describe them using phrases like:

"She's got great judgement."

"He's someone I trust with difficult decisions."

"Give it to her. She'll figure it out."

"He's the person everyone asks when things get complicated."

Notice what these descriptions have in common. They do not praise programming ability, or AI expertise, or knowledge of a particular technology. They describe something much harder to define. The ability to think well in complex situations.

Looking Beneath the Surface

So what explains this ability to think well in complex situations? The obvious explanation is that these engineers simply know more. But knowledge alone does not seem sufficient. Two engineers can possess similar technical knowledge and yet arrive at very different decisions.

The difference is not merely what they know. It appears to be in how they think. More specifically, it is how they construct an understanding of a situation before taking a decision.

In simple terms, we call this judgement, and in this context *Engineering Judgement*. It is the quality of thinking that shapes how engineers understand a situation and decide what to do.

The Thinking Beneath Engineering

Engineering judgement is not another skill sitting alongside programming, architecture, communication or even leadership. It is something more fundamental. It is the quality of thinking that connects all these skills.

The pages that follow explore a simple idea:

The most important engineering decisions are rarely determined by technical knowledge alone. They are shaped by the quality of thinking that engineers bring to complex, uncertain, and evolving situations.

Understanding that thinking, and learning how to strengthen it may be one of the most valuable investments an engineering organization can make.

CHAPTER 2

The Role of Judgement in Engineering

Ask an engineering leader what their team needs to improve, and the answers are remarkably consistent, regardless of the organization, technology stack, or industry.

"We need better estimation."

"Our architecture decisions aren't consistent."

"We struggle with technical debt."

"Our engineers need to communicate better."

"We're finding AI-generated code difficult to debug."

Each of these appears to describe a different problem. So organizations search for different solutions: an estimation workshop, a software architecture course, or stakeholder communication training. They invest in tools and processes: an AI coding policy or an incident management process.

Yet over time a pattern begins to emerge. The same engineers perform well across many of these situations. Likewise, the same engineers struggle across several of them. That observation raises a curious question:

Are these really separate problems? Or are they different expressions of the same underlying challenge?

Consider Estimation

Few activities create as much frustration inside engineering teams as software estimation. Managers ask for dates. Teams debate story points. Engineers wrestle with uncertainty before eventually committing to an estimate they may not feel confident about.

Then the project begins. Dependencies emerge, requirements shift and unexpected complexities appear. The estimate begins to unravel piece by piece.

When an estimate falters, teams focus on what went wrong: what was missed, what changed, and what turned out to be more difficult than expected. But the deeper issue may be different.

It could be how the team thought about the work. They underestimated uncertainty, overlooked assumptions, missed important interactions, or focused on individual components without considering the system as a whole.

These problems reflect how engineers understand situations, reason under uncertainty, and make judgements when the full picture is not yet visible. The issue, then, is not estimation alone. It is judgement.

Consider Architecture

Architecture decisions appear highly technical. They involve system design, technology choices, performance, scalability, resilience and maintainability. Yet experienced architects know that technical knowledge alone rarely determines the best decision.

Architecture involves balancing competing concerns, understanding organizational context, considering future consequences, reconciling different stakeholder needs, and accepting that every design decision introduces new trade-offs.

Again, the central challenge is not simply technical expertise. It is judgement.

Consider Production Incidents

Production incidents place engineers under intense pressure. Information is incomplete. Customers are affected. Business leaders want updates. Teams must make decisions quickly: Restore service immediately or continue investigating? Rollback or apply a temporary workaround? Escalate, or manage the situation within the team?

Every decision carries consequences. The challenge is rarely only about finding a technical solution. It is deciding which action makes most sense given the situation. Again, judgement.

Consider AI

Artificial intelligence has introduced new capabilities into software engineering at remarkable speed. Engineers can now use AI to generate code and tests, while also getting suggestions for architecture and debugging within seconds. The technical capability is impressive. But the engineering challenge has changed rather than disappeared.

How trustworthy is the AI recommendation? What assumptions did the model make? What important context might it have missed? And once we start using AI regularly, a broader question emerges: how much should we rely on it? When should AI be trusted, questioned, or ignored?

These are not questions about AI prompts. They are questions of judgement.

The Same Pattern Repeats

Once you see this pattern, it becomes difficult to ignore. Estimation, architecture, technical debt, prioritization, delegation, production incidents, AI adoption — each situation appears different. Yet beneath every situation lies a remarkably similar challenge.

The engineer must construct an understanding of an incomplete situation. Interpret imperfect evidence. Recognize assumptions. Consider multiple perspectives. Anticipate future consequences. Act despite uncertainty. Learn from the outcome.

The situations change. The underlying thinking remains surprisingly consistent.

What It Suggests

This raises an interesting question:

What kind of thinking consistently produces good engineering decisions across all of these situations?

That question shifts the conversation. Instead of treating each engineering challenge as an isolated competency, it invites us to look beneath the surface and search for the thinking that they all have in common.

That search is where the idea of engineering judgement truly begins.

CHAPTER 3

Why Judgement Takes More Than Experience

Engineering organizations invest heavily in developing technical capability. New engineers learn programming languages, frameworks, testing strategies, and dozens of other skills. As engineers grow, organizations invest again. This time in architecture workshops, leadership programs, and management training.

Modern software engineering demands continuous learning, and rightly so. Yet there is one subject that receives little deliberate attention: Judgement.

We Assume Judgement Will Somehow Emerge

Ask an engineering leader how engineers develop architectural judgement. The typical answers are:

"They'll learn over time."

"They'll gain from experience."

"They'll eventually figure it out."

These answers are understandable. After all, judgement cannot simply be taught through a presentation or a checklist. It grows slowly. It depends on context. It develops through experience. But this creates an interesting paradox.

Organizations deliberately develop almost every important engineering capability except the one that seems to connect all the others. Instead, judgement is often treated as something that simply appears after enough years in the profession.

Experience Is Essential, But Unpredictable

Experience matters. Every production incident teaches something new, every architectural decision reveals new trade-offs, and every stakeholder conversation exposes new

perspectives. Experience broadens an engineer's view of the profession. But it does not affect everyone in the same way.

Two engineers can participate in the same project. One becomes noticeably wiser. The other simply grows older. Two managers can lead equally complex teams. One develops increasingly thoughtful judgement. The other repeats the same mistakes year after year.

The experience was similar. The development was not. Something happened between the experience and the learning.

Knowledge Has Limits

Knowledge remains one of the great strengths of software engineering. But knowledge alone rarely answers the most important engineering questions.

Should we rewrite the system?

Should we delay the release?

Should we trust the AI-generated solution?

These questions rarely have universally correct answers. They depend on context, constraints, trade-offs, and consequences that cannot be known with certainty. Knowledge informs these decisions. It does not make them.

Frameworks Help, But Do Not Replace Judgement

Software engineering has no shortage of frameworks. It has decision matrices, prioritization models, architecture principles, planning techniques and checklists for every task you can think of. Each exists for good reason.

Frameworks help organize thinking. They reduce oversight and encourage consistency. Used thoughtfully, they are valuable. Yet every experienced engineering leader has encountered situations where following the framework was not enough.

The framework did not fail. It simply reached the boundary of what frameworks can do. At that point, the quality of thinking becomes more important than the quality of the process.

AI Makes the Conversation Even More Important

Artificial intelligence has accelerated many aspects of software development. Generating code has become faster. Exploring design alternatives has become simpler. Acquiring technical knowledge has become easier.

As knowledge becomes easier to acquire, it stops being a competitive advantage. What increasingly differentiates engineers is not what they know, but how they think.

AI can generate code, and suggest design alternatives. But it cannot fully understand an organization's unique context, weigh competing priorities, or decide what level of risk is acceptable. It can answer questions, but it cannot determine which questions matter most.

As AI takes on more of the work of generating and processing information, the quality of judgement becomes an even greater differentiator.

Perhaps We Are Asking the Wrong Question

The traditional question has been:

"How do we teach engineers more?"

A more pertinent question might be:

"How can engineers learn to think more effectively about complex engineering situations?"

That is a different question altogether. It shifts attention from transferring knowledge towards strengthening the thinking that converts knowledge into action. And that is what we will explore next.

CHAPTER 4

The Thinking Behind Sound Judgement

Imagine sitting quietly in the corner of an engineering organization for a year. You attend architecture reviews, sprint planning sessions and production incident calls. You observe hiring interviews, design discussions and leadership meetings.

One by one, situations begin to accumulate. Initially, they appear unrelated. Different teams, different personalities, different business problems and different decisions. But after enough observation, something interesting begins to happen.

The problems look different on the surface, but the thinking required to navigate them starts to look remarkably similar.

Looking Beyond the Decision

Consider a team estimating a software project. At first glance, this appears to be a discussion about effort and dates. But listen more carefully.

One engineer asks, *"What assumptions are we making?"*

Another says, *"We've built something similar before."*

Someone else points out, *"What happens if this service isn't ready?"*

Notice what is happening. The discussion is not just about estimation. It is about examining evidence, recognizing patterns, questioning assumptions, understanding relationships and thinking ahead.

A similar pattern appears during an architectural discussion when engineers ask,

"How will this affect operations?"

"What happens when traffic doubles?"

"Are we optimizing for today's problem or next year's?"

During a production incident,

"Do we actually know the root cause?"

"What evidence do we have?"

"What has changed?"

Different situations. Remarkably similar thinking.

The Questions Beneath the Questions

Engineering discussions appear to revolve around technical answers. Yet beneath those answers lie a surprisingly consistent set of questions.

What do we actually know?

What assumptions are we making?

What perspectives have we not considered?

What pattern does this resemble?

What might happen next?

How confident should we be?

These questions are not specific to architecture, estimation or AI. They appear whenever engineering decisions must be made.

The Ten Thinking Mechanisms

When these recurring questions are examined more closely, they appear to arise from a relatively small number of underlying ways of thinking. The same ten foundational thinking mechanisms appear again and again.

1. **Evidence Evaluation** – Distinguishing reliable evidence from assumptions, opinions and incomplete information.
2. **Pattern Recognition** – Recognizing recurring structures, behaviours and relationships across different situations.

3. **Assumption Awareness** – Identifying the assumptions that quietly shape decisions before they become hidden risks.
4. **Systems Thinking** – Understanding how components, people, and decisions interact within a larger system.
5. **Perspective Integration** – Bringing together multiple legitimate viewpoints before forming a conclusion.
6. **Contextual Reasoning** – Understanding what matters in a particular situation instead of applying general rules mechanically.
7. **Long-Term Thinking** – Considering the future consequences of today's technical and organizational decisions.
8. **Thinking Under Uncertainty** – Making thoughtful decisions despite incomplete information while remaining aware of uncertainty.
9. **Reflection** – Learning deliberately from previous experiences and decisions.
10. **Cognitive Flexibility** – Revising existing understanding when new evidence or perspectives emerge.

Individually, none of these mechanisms are remarkable. Together, they form the foundation of better thinking—a practical compass for understanding complex situations, navigating uncertainty, and making sound decisions.

Looking One Layer Deeper

As we move beneath engineering situations, and identify recurring ways of thinking that support sound engineering decisions, another question emerges.

If these thinking mechanisms consistently appear across engineering situations, what exactly are they improving? What changes inside an engineer as their thinking becomes stronger?

That is the question we'll answer next.

CHAPTER 5

How Thinking Shapes Judgement

Imagine asking two experienced engineers to respond to the same problem. Both have similar technical backgrounds, worked on similar technologies and have access to the same production metrics.

Yet after an hour of discussion, they recommend very different courses of action. One wants to rewrite the system. The other prefers incremental improvement. Why?

The information available to both engineers was almost identical. The difference lies somewhere else. Not in the facts. But in how those facts were understood.

Thinking Builds Internal Models

Whether engineers realize it or not, they are continuously constructing internal representations of the situations they face. These representations help them make sense of complexity by connecting experience, observations, assumptions, constraints, relationships, and expectations into a coherent picture.

That picture helps them answer the question:

"Given everything I currently understand, what is the most sensible thing to do?"

This internal picture is called a *mental model*.

Facts matter. Experience matters. Knowledge matters. But none of them determines judgement on their own. They first become part of a mental model. The quality of that model then shapes the quality of the decision.

Five Qualities of Strong Mental Models

Looking across many engineering situations, strong mental models tend to share five important qualities.

1. **Accuracy** – They correspond closely to reality. They distinguish observations from assumptions, evidence from speculation, and facts from interpretation.

2. **Completeness** – They include the important parts of the situation rather than the most visible details. They incorporate systems, stakeholders, context, dependencies and constraints.
3. **Projection** – They help engineers anticipate what is likely to happen next. They allow future consequences, second-order effects, and long-term trade-offs to become visible before decisions are made.
4. **Adaptability** – They continue evolving as new evidence appears. Rather than defending earlier conclusions, engineers update their understanding when reality changes.
5. **Calibration** – They accurately reflect the level of confidence the situation deserves. Strong engineers know not only what they understand. They also recognize what they do not understand.

Together, these qualities determine how well a mental model supports thinking when situations become complex or uncertain.

Strong mental models help engineers see what matters, anticipate what might happen, recognize what remains uncertain, and adapt as reality unfolds. They turn experience and knowledge into a more reliable basis for judgement.

Judgement Emerges from Mental Models

The foundational thinking mechanisms we explored in the previous chapter strengthen these mental models in different ways.

Evidence evaluation improves accuracy. Systems thinking improves completeness. Long-term thinking strengthens projection. Reflection improves adaptability. Thinking under uncertainty improves calibration.

Different thinking mechanisms contribute to different aspects of the mental model. Together, they improve the quality of judgement itself.

A Different Goal for Engineering Development

When you understand this perspective, the goal of engineering development changes.

Instead of asking,

"How do we create better estimators or how do we produce better architects?"

we start asking a more fundamental question:

How do we help engineers build more capable mental models of the complex situations they encounter every day?

That question shifts the focus away from individual competencies towards the quality of thinking that underlines them all. That is where we turn to next.

CHAPTER 6

How Judgement Develops

Mental models rarely change simply with time and experience. They change when engineers deliberately examine the thinking behind their experiences—questioning assumptions, revisiting conclusions, considering alternative explanations, and paying attention when evidence challenges what they already believe. Sometimes, it happens when someone asks a question that had never previously occurred to them.

These moments often feel small. A conversation after a difficult meeting. A quiet realization during a postmortem. A colleague wondering, "Why were we so confident?" A production issue that refuses to fit the original explanation. Each becomes an opportunity to reconstruct how a situation is understood.

Over time, these small shifts accumulate. Eventually the engineer is no longer merely solving engineering problems differently. They are thinking differently.

For some engineers, this process happens naturally. They learn from experience by questioning, reflecting, and gradually refining how they understand situations. For others, experience accumulates without the same deliberate examination, and their mental models remain largely unchanged.

The Ten Transformational Processes

Looking across many situations, certain developmental processes appear repeatedly. Not as techniques or checklists, but as recurring ways in which engineers can gradually strengthen the quality of their thinking.

1. **Encountering Reality** in ways that challenge existing assumptions.
2. **Noticing Yourself** to become more aware of your own thinking while decisions are being made.
3. **Examining Your Thinking** to understand why particular conclusions feel obvious.
4. **Reflection** on important experiences.

5. **Inquiry** driven exploration of situations rather than immediate answers.
6. **Decision Reconstruction** to understand the reasoning behind them.
7. **Perspective Expansion** beyond your own technical viewpoint.
8. **Exploring Possibilities** before committing to a solution.
9. **Holding the Question** or remaining with uncertainty long enough for deeper understanding to emerge.
10. **Zooming In and Out** by moving repeatedly between the finer details and the broader system.

These processes rarely occur in isolation. A difficult architectural decision will involve several of them simultaneously. So will a production incident or a career transition.

Each process reshapes the engineer's mental models. Not by adding more information, but by changing how information is interpreted.

Judgement Develops Through Deliberate Thinking

Seen from this perspective, engineering judgement becomes easier to understand.

It is not a mysterious talent possessed by a fortunate few. Nor is it simply experience accumulated over time. It is the gradual consequence of repeatedly examining, challenging, expanding, and refining the mental models that guide engineering decisions.

The engineers who consistently questions their assumptions or revisit difficult decisions to understand the reasoning behind them are gradually changing how they think. So are the ones who routinely considers the perspectives of operations, customers and product managers.

These changes may appear small in isolation. Across an entire career, they become profound.

The Role of Coaching

If judgement develops through transforming the thinking behind engineering decisions, then the role of coaching becomes easier to understand.

A coach cannot provide twenty years of engineering experience. Nor can a coach make design decisions on behalf of an engineer. What coaching can do is something different.

It can create conversations that *make thinking visible*. It can help engineers notice assumptions they had not recognized. Question conclusions that felt obvious. Consider perspectives they had overlooked. Remain with uncertainty instead of rushing toward premature certainty. Reflect more deeply on the reasoning behind important decisions.

In other words, coaching does not replace experience. It helps engineers learn more deliberately from experience. That distinction may be the most important contribution coaching can make to the development of engineering judgement.

CHAPTER 7

Developing Engineering Judgement

If engineering judgement develops through the gradual improvement of mental models, an important implication follows. Judgement can be developed. And it can be developed intentionally.

That idea challenges one of the oldest assumptions in engineering. That judgement is simply something engineers acquire if they remain in the profession long enough.

Experience matters because it creates opportunities for learning. But it does not guarantee learning. What matters is whether experience changes the way an engineer thinks. Experience is not the destination. It is the raw material.

Learning to See Thinking

One of the most important transitions in professional growth occurs when engineers begin paying attention not only to their decisions, but also to the thinking that produced those decisions.

Instead of asking, *"Did I make the right decision?"*

they start asking, *"How did I arrive at that decision?"*

Instead of asking, *"What should I do?"*

they start asking, *"How should I think about this?"*

These questions produce better thinking which gradually produces better judgement.

Deliberate Development

If judgement develops through transforming the quality of thinking, then organizations can create environments where that transformation becomes more likely.

Engineering leaders can encourage thoughtful decision reviews rather than outcome reviews alone. Teams can examine assumptions rather than only tasks. Retrospectives can explore reasoning instead of merely documenting events.

Architecture discussions can value thoughtful questions as much as confident answers. Production incidents can become opportunities to improve understanding rather than simply restore service. AI can become a partner in exploring alternatives rather than a substitute for independent thinking.

These practices give engineers opportunities to strengthen their thinking. Together, they create conditions in which sound judgement is more likely to develop.

Why Coaching Fits Naturally

This perspective also clarifies the contribution of coaching.

Coaching is often misunderstood as providing advice, sharing experience, or helping engineers solve difficult problems. Those activities can be valuable, but they are not the unique contribution of coaching. Coaching creates something different.

It creates a space where engineers can examine their own thinking. A space where assumptions become visible. Where conclusions are questioned without being attacked. Where alternative perspectives can be explored without needing immediate agreement.

It creates a space where uncertainty can be held long enough for deeper understanding to emerge. Where important decisions can be revisited to understand the reasoning that produced them.

These conversations create small but meaningful changes in how engineers think. Those changes accumulate, and over time, reshape judgement itself.

CHAPTER 8

Developing the Engineers We Need

Every profession has ideas that shape how it develops its people. Medicine complements medical knowledge with clinical judgement. Aviation complements flying skills with sound decision-making under pressure. The military complements tactical expertise with command judgement.

In the same way, software engineering must complement technical knowledge with another capability: engineering judgement.

The Future of Software Engineering

Artificial intelligence will continue changing software engineering, reshaping how engineers build, operate, and evolve systems. Many of today's technical practices will be replaced by better ones. Yet some aspects of engineering will remain unchanged.

Engineers will still need to understand situations that are incomplete. They will have to balance short-term delivery with long-term consequences. They will need to navigate disagreements, and make decisions whose outcomes are not known in advance.

Perhaps the greatest opportunity created by AI is not just faster software development. It is the opportunity to pay greater attention to the distinctly human aspects of engineering: understanding, reasoning, discernment, perspective, and judgement.

Developing Better Engineers

Throughout this work, one idea has gradually emerged. Engineering judgement is neither a mysterious talent nor an inevitable consequence of experience. It is the result of strengthening the mental models engineers use to understand complex situations.

Those mental models improve through deliberate ways of thinking. They improve through experience that is examined rather than merely accumulated. They improve through thoughtful questions rather than quick answers. And, they improve when assumptions

become visible, perspectives broaden, uncertainty is respected, and learning continues long after a decision has been made.

Seen this way, engineering judgement is not just another competency to add to the many skills organizations teach their engineers. It is the thread that connects them all.

An Invitation

Perhaps the most important question is no longer,

"How do we build better software?"

It is,

"How do we develop engineers who consistently make better engineering decisions?"

That question deserves far more attention than it has traditionally received. Not because engineering needs fewer technical experts. It needs more.

But because the complexity of modern engineering increasingly demands something alongside expertise. Thoughtful judgement.

That is how the next generation of engineering organizations will distinguish themselves. Not simply through the technologies they adopt. But through the quality of thinking they cultivate.

Because, in the end, software engineering is not just the discipline of building systems. It is also the discipline of making thoughtful decisions about the systems we choose to build.

And the most enduring investment an engineering organization can make is not only in better technology. It is in developing better judgement.

Thank you for reading.

PARAG CHITNIS

paragchitnis.com