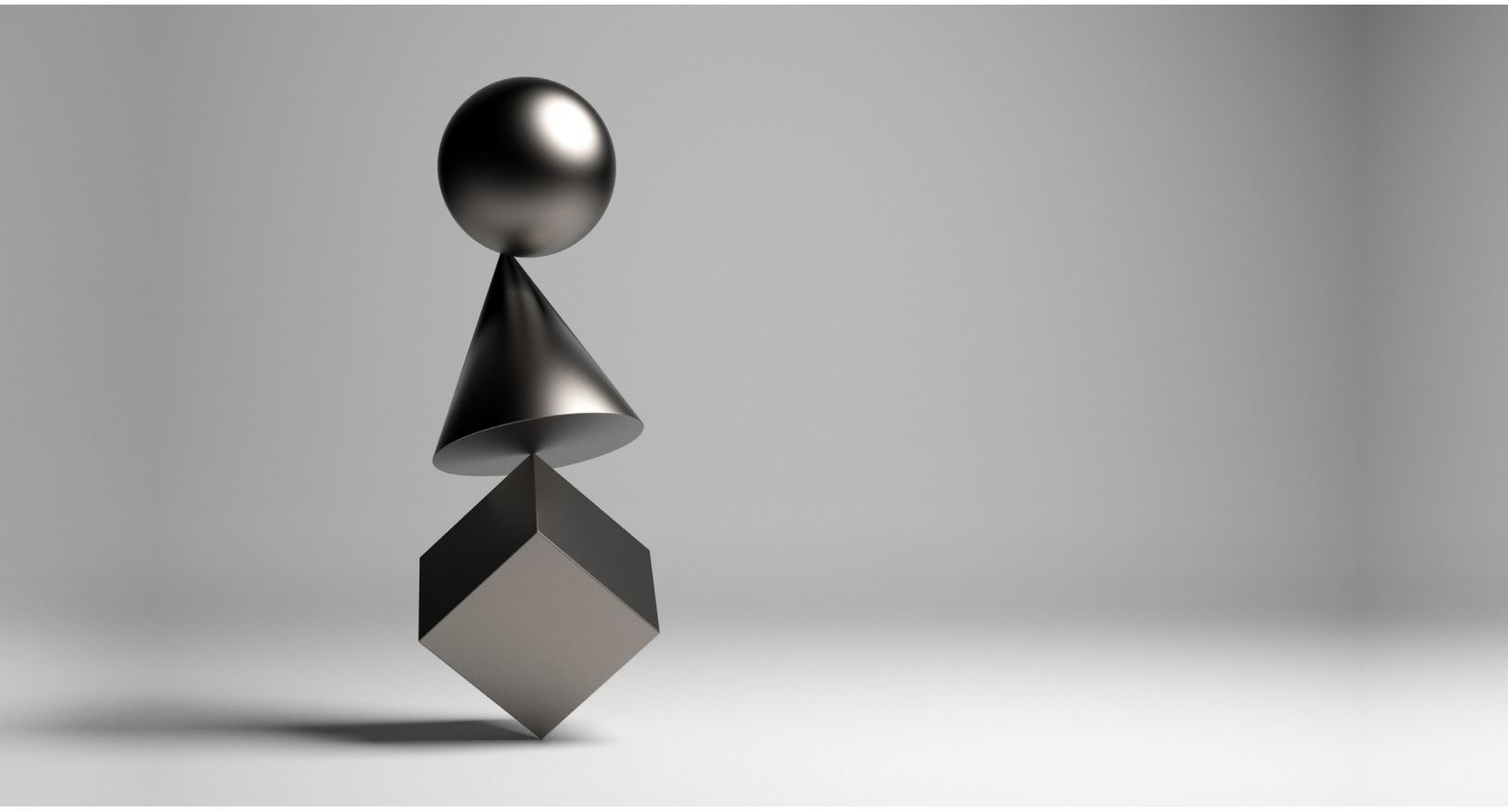
McKinsey & Company

Strategy & Corporate Finance Practice

How to master the seven-step problem-solving process

Structured problem solving can be used to address almost any complex challenge in business or public policy.



September 2019

In this episode of the *McKinsey Podcast*, Simon London speaks with Charles Conn, CEO of venture-capital firm Oxford Sciences Innovation, and McKinsey senior partner Hugo Sarrazin about the complexities of different problem-solving strategies.

Podcast transcript

Simon London: Hello, and welcome to this episode of the *McKinsey Podcast,* with me, Simon London. What's the number-one skill you need to succeed professionally? Salesmanship, perhaps? Or a facility with statistics? Or maybe the ability to communicate crisply and clearly? Many would argue that at the very top of the list comes problem solving: that is, the ability to think through and come up with an optimal course of action to address any complex challenge—in business, in public policy, or indeed in life.

Looked at this way, it's no surprise that McKinsey takes problem solving very seriously, testing for it during the recruiting process and then honing it, in McKinsey consultants, through immersion in a structured seven-step method. To discuss the art of problem solving, I sat down in California with McKinsey senior partner Hugo Sarrazin and also with Charles Conn. Charles is a former McKinsey partner, entrepreneur, executive, and coauthor of the book *Bulletproof Problem Solving: The One Skill That Changes Everything* [John Wiley & Sons, 2018].

Charles and Hugo, welcome to the podcast. Thank you for being here.

Hugo Sarrazin: Our pleasure.

Charles Conn: It's terrific to be here.

Simon London: Problem solving is a really interesting piece of terminology. It could mean so many different things. I have a son who's a teenage climber. They talk about solving problems. Climbing is problem solving. Charles, when you talk about problem solving, what are you talking about?

Charles Conn: For me, problem solving is the answer to the question "What should I do?" It's interesting when there's uncertainty and complexity, and when it's meaningful because there are consequences. Your son's climbing is a perfect example. There are consequences, and it's complicated, and there's uncertainty—can he make that grab? I think we can apply that same frame almost at any level. You can think about questions like "What town would I like to live in?" or "Should I put solar panels on my roof?"

You might think that's a funny thing to apply problem solving to, but in my mind it's not fundamentally different from business problem solving, which answers the question "What should my strategy be?" Or problem solving at the policy level: "How do we combat climate change?" "Should I support the local school bond?" I think these are all part and parcel of the same type of question, "What should I do?"

I'm a big fan of structured problem solving. By following steps, we can more clearly understand what problem it is we're solving, what are the components of the problem that we're solving, which components are the most important ones for us to pay attention to, which analytic techniques we should apply to those, and how we can synthesize what we've learned back into a compelling story. That's all it is, at its heart.

I think sometimes when people think about seven steps, they assume that there's a rigidity to this. That's not it at all. It's actually to give you the scope for creativity, which often doesn't exist when your problem solving is muddled.

Simon London: You were just talking about the seven-step process. That's what's written down in the book, but it's a very McKinsey process as well. Without getting too deep into the weeds, let's go through the steps, one by one. You were just talking about problem definition as being a particularly important thing to get right first. That's the first step. Hugo, tell us about that.

Hugo Sarrazin: It is surprising how often people jump past this step and make a bunch of assumptions. The most powerful thing is to step back and ask the basic questions—"What are we trying to solve? What are the constraints that exist? What are the dependencies?" Let's make those explicit and really push the thinking and defining. At McKinsey, we spend an enormous amount of time in writing that little statement, and the statement, if you're a logic purist, is great. You debate. "Is it an 'or'? Is it an 'and'? What's the action verb?" Because all these specific words help you get to the heart of what matters.

Simon London: So this is a concise problem statement.

Hugo Sarrazin: Yeah. It's not like "Can we grow in Japan?" That's interesting, but it is "What, specifically, are we trying to uncover in the growth of a product in Japan? Or a segment in Japan? Or a channel in Japan?" When you spend an enormous amount of time, in the first meeting of the different stakeholders, debating this and having different people put forward what they think the problem definition is, you realize that people have completely different views of why they're here. That, to me, is the most important step.

Charles Conn: I would agree with that. For me, the problem context is critical. When we understand "What are the forces acting upon your decision maker? How quickly is the answer needed? With what precision is the answer needed? Are there areas that are off limits or areas where we would particularly like to find our solution? Is the decision maker open to exploring other areas?" then you not only become more efficient, and move toward what we call the critical path in problem solving, but you also make it so much more likely that you're not going to waste your time or your decision maker's time.

How often do especially bright young people run off with half of the idea about what the problem is and start collecting data and start building models—only to discover that they've really gone off half-cocked.

Hugo Sarrazin: Yeah.

Charles Conn: And in the wrong direction.

Hugo Sarrazin: Yeah.

Simon London: OK. So step one—and there is a real art and a structure to it—is define the problem. Step two, Charles?

Charles Conn: My favorite step is step two, which is to use logic trees to disaggregate the problem. Every problem we're solving has some complexity and some uncertainty in it. The only way that we can really get our team working on the problem is to take the problem apart into logical pieces.

What we find, of course, is that the way to disaggregate the problem often gives you an insight into the answer to the problem quite quickly. I love to do two or three different cuts at it, each one giving a bit of a different insight into what might be going wrong. By doing sensible disaggregations, using logic trees, we can figure out which parts of the problem we should be looking at, and we can assign those different parts to team members.

Simon London: What's a good example of a logic tree on a sort of ratable problem?

Charles Conn: Maybe the easiest one is the classic profit tree. Almost in every business that I would take a look at, I would start with a profit or return-on-assets tree. In its simplest form, you have the components of revenue, which are price and quantity, and the components of cost, which are cost and quantity. Each of those can be broken out. Cost can be broken into variable cost and fixed cost. The components of price can be broken into what your pricing scheme is. That simple tree often provides insight into what's going on in a business

or what the difference is between that business and the competitors.

If we add the leg, which is "What's the asset base or investment element?"—so profit divided by assets—then we can ask the question "Is the business using its investments sensibly?" whether that's in stores or in manufacturing or in transportation assets. I hope we can see just how simple this is, even though we're describing it in words.

When I went to work with Gordon Moore at the Moore Foundation, the problem that he asked us to look at was "How can we save Pacific salmon?" Now, that sounds like an impossible question, but it was amenable to precisely the same type of disaggregation and allowed us to organize what became a 15-year effort to improve the likelihood of good outcomes for Pacific salmon.

Simon London: Now, is there a danger that your logic tree can be impossibly large? This, I think, brings us onto the third step in the process, which is that you have to prioritize.

Charles Conn: Absolutely. The third step, which we also emphasize, along with good problem definition, is rigorous prioritization—we ask the questions "How important is this lever or this branch of the tree in the overall outcome that we seek to achieve? How much can I move that lever?" Obviously, we try and focus our efforts on ones that have a big impact on the problem and the ones that we have the ability to change. With salmon, ocean conditions turned out to be a big lever, but not one that we could adjust. We focused our attention on fish habitats and fish-harvesting practices, which were big levers that we could affect.

People spend a lot of time arguing about branches that are either not important or that none of us can change. We see it in the public square. When we deal with questions at the policy level—"Should you support the death penalty?" "How do we affect climate change?" "How can we uncover the causes and address homelessness?"—it's even more important that we're focusing on levers that are big and movable.

Simon London: Let's move swiftly on to step four. You've defined your problem, you disaggregate it, you prioritize where you want to analyze—what you want to really look at hard. Then you got to the work plan. Now, what does that mean in practice?

Hugo Sarrazin: Depending on what you've prioritized, there are many things you could do. It could be breaking the work among the team members so that people have a clear piece of the work to do. It could be defining the specific analyses that need to get done and executed, and being clear on time lines. There's always a level-one answer, there's a level-two answer, there's a level-three answer. Without being too flippant, I can solve any problem during a good dinner with wine. It won't have a whole lot of backing.

Simon London: Not going to have a lot of depth to it.

Hugo Sarrazin: No, but it may be useful as a starting point. If the stakes are not that high, that could be OK. If it's really high stakes, you may need level three and have the whole model validated in three different ways. You need to find a work plan that reflects the level of precision, the time frame you have, and the stakeholders you need to bring along in the exercise.

Charles Conn: I love the way you've described that, because, again, some people think of problem solving as a linear thing, but of course what's critical is that it's iterative. As you say, you can solve the problem in one day or even one hour.

Hugo Sarrazin: Yeah.

Charles Conn: We encourage our teams everywhere to do that. We call it the one-day answer or the one-hour answer. In work planning, we're always iterating. Every time you see a 50-page work plan that stretches out to three months, you know it's wrong. It will be outmoded very quickly by that learning process that you described. Iterative problem solving is a critical part of this. Sometimes, people think work planning sounds dull, but it isn't. It's how we know what's

expected of us and when we need to deliver it and how we're progressing toward the answer. It's also the place where we can deal with biases. Bias is a feature of every human decision-making process. If we design our team interactions intelligently, we can avoid the worst sort of biases.

Simon London: Here we're talking about cognitive biases primarily, right? It's not that I'm biased against you because of your accent or something. These are the cognitive biases that behavioral sciences have shown we all carry around, things like anchoring, overoptimism—these kinds of things.

Both: Yeah.

Charles Conn: Availability bias is the one that I'm always alert to. You think you've seen the problem before, and therefore what's available is your previous conception of it—and we have to be most careful about that. In any human setting, we also have to be careful about biases that are based on hierarchies, sometimes called sunflower bias. I'm sure, Hugo, with your teams, you make sure that the youngest team members speak first. Not the oldest team members, because it's easy for people to look at who's senior and alter their own creative approaches.

Hugo Sarrazin: It's helpful, at that moment—if someone is asserting a point of view—to ask the question "This was true in what context?" You're trying to apply something that worked in one context to a different one. That can be deadly if the context has changed, and that's why organizations struggle to change. You promote all these people because they did something that worked well in the past, and then there's a disruption in the industry, and they keep doing what got them promoted even though the context has changed.

Simon London: Right. Right.

Hugo Sarrazin: So it's the same thing in problem solving.

Charles Conn: And it's why diversity in our teams is so important. It's one of the best things about the world that we're in now. We're likely to have people from different socioeconomic, ethnic, and national backgrounds, each of whom sees problems from a slightly different perspective. It is therefore much more likely that the team will uncover a truly creative and clever approach to problem solving.

Simon London: Let's move on to step five. You've done your work plan. Now you've actually got to do the analysis. The thing that strikes me here is that the range of tools that we have at our disposal now, of course, is just huge, particularly with advances in computation, advanced analytics. There's so many things that you can apply here. Just talk about the analysis stage. How do you pick the right tools?

Charles Conn: For me, the most important thing is that we start with simple heuristics and explanatory statistics before we go off and use the big-gun tools. We need to understand the shape and scope of our problem before we start applying these massive and complex analytical approaches.

Simon London: Would you agree with that?

Hugo Sarrazin: I agree. I think there are so many wonderful heuristics. You need to start there before you go deep into the modeling exercise. There's an interesting dynamic that's happening, though. In some cases, for some types of problems, it is even better to set yourself up to maximize your learning. Your problem-solving methodology is test and learn, test and learn, test and learn, and iterate. That is a heuristic in itself, the A/B testing that is used in many parts of the world. So that's a problem-solving methodology. It's nothing different. It just uses technology and feedback loops in a fast way. The other one is exploratory data analysis. When you're dealing with a large-scale problem, and there's so much data, I can get to the heuristics that Charles was talking about through very clever visualization of data.

You test with your data. You need to set up an environment to do so, but don't get caught up in neural-network modeling immediately. You're testing, you're checking—"Is the data right? Is it sound? Does it make sense?"—before you launch too far.

Simon London: You do hear these ideas—that if you have a big enough data set and enough algorithms, they're going to find things that you just wouldn't have spotted, find solutions that maybe you wouldn't have thought of. Does machine learning sort of revolutionize the problem-solving process? Or are these actually just other tools in the toolbox for structured problem solving?

Charles Conn: It can be revolutionary. There are some areas in which the pattern recognition of large data sets and good algorithms can help us see things that we otherwise couldn't see. But I do think it's terribly important we don't think that this particular technique is a substitute for superb problem solving, starting with good problem definition. Many people use machine learning without understanding algorithms that themselves can have biases built into them. Just as 20 years ago, when we were doing statistical analysis, we knew that we needed good model definition, we still need a good understanding of our algorithms and really good problem definition before we launch off into big data sets and unknown algorithms.

Simon London: Step six. You've done your analysis.

Charles Conn: I take six and seven together, and this is the place where young problem solvers often make a mistake. They've got their analysis, and they assume that's the answer, and of course it isn't the answer. The ability to synthesize the pieces that came out of the analysis and begin to weave those into a story that helps people answer the question "What should I do?" This is back to where we started. If we can't synthesize, and we can't tell a story, then our decision maker can't find the answer to "What should I do?"

Simon London: But, again, these final steps are about motivating people to action, right?

Charles Conn: Yeah.

Simon London: I am slightly torn about the nomenclature of problem solving because it's on paper, right? Until you motivate people to action, you actually haven't solved anything.

Charles Conn: I love this question because I think decision-making theory, without a bias to action, is a waste of time. Everything in how I approach this is to help people take action that makes the world better.

Simon London: Hence, these are absolutely critical steps. If you don't do this well, you've just got a bunch of analysis.

Charles Conn: We end up in exactly the same place where we started, which is people speaking across each other, past each other in the public square, rather than actually working together, shoulder to shoulder, to crack these important problems.

Simon London: In the real world, we have a lot of uncertainty—arguably, increasing uncertainty. How do good problem solvers deal with that?

Hugo Sarrazin: At every step of the process. In the problem definition, when you're defining the context, you need to understand those sources of uncertainty and whether they're important or not important. It becomes important in the definition of the tree.

You need to think carefully about the branches of the tree that are more certain and less certain as you define them. They don't have equal weight just because they've got equal space on the page. Then, when you're prioritizing, your prioritization approach may put more emphasis on things that have low probability but huge impact—or, vice versa, may put a lot of priority on things that are very likely and, hopefully, have a reasonable impact. You can introduce that along the way. When you come back to the synthesis, you just need to be

nuanced about what you're understanding, the likelihood.

Often, people lack humility in the way they make their recommendations: "This is the answer." They're very precise, and I think we would all be well-served to say, "This is a likely answer under the following sets of conditions" and then make the level of uncertainty clearer, if that is appropriate. It doesn't mean you're always in the gray zone; it doesn't mean you don't have a point of view. It just means that you can be explicit about the certainty of your answer when you make that recommendation.

Simon London: So it sounds like there is an underlying principle: "Acknowledge and embrace the uncertainty. Don't pretend that it isn't there. Be very clear about what the uncertainties are up front, and then build that into every step of the process."

Hugo Sarrazin: Every step of the process.

Simon London: Yeah. We have just walked through a particular structured methodology for problem solving. But, of course, this is not the only structured methodology for problem solving. One that is also very well-known is design thinking, which comes at things very differently. So, Hugo, I know you have worked with a lot of designers. Just give us a very quick summary. Design thinking—what is it, and how does it relate?

Hugo Sarrazin: It starts with an incredible amount of empathy for the user and uses that to define the problem. It does pause and go out in the wild and spend an enormous amount of time seeing how people interact with objects, seeing the experience they're getting, seeing the pain points or joy—and uses that to infer and define the problem.

Simon London: Problem definition, but out in the world.

Hugo Sarrazin: With an enormous amount of empathy. There's a huge emphasis on empathy. Traditional, more classic problem solving is you define the problem based on an understanding of the situation. This one almost presupposes that we don't know the problem until we go see it. The second thing is you need to come up with multiple scenarios or answers or ideas or concepts, and there's a lot of divergent thinking initially. That's slightly different, versus the prioritization, but not for long. Eventually, you need to kind of say, "OK, I'm going to converge again." Then you go and you bring things back to the customer and get feedback and iterate. Then you rinse and repeat, rinse and repeat. There's a lot of tactile building, along the way, of prototypes and things like that. It's very iterative.

Simon London: So, Charles, are these complements or are these alternatives?

Charles Conn: I think they're entirely complementary, and I think Hugo's description is perfect. When we do problem definition well in classic problem solving, we are demonstrating the kind of empathy, at the very beginning of our problem, that design thinking asks us to approach. When we ideate—and that's very similar to the disaggregation, prioritization, and work-planning steps—we do precisely the same thing, and often we use contrasting teams, so that we do have divergent thinking. The best teams allow divergent thinking to bump them off whatever their initial biases in problem solving are. For me, design thinking gives us a constant reminder of creativity, empathy, and the tactile nature of problem solving, but it's absolutely complementary, not alternative.

Simon London: I think, in a world of cross-functional teams, an interesting question is do people with design-thinking backgrounds really work well together with classical problem solvers? How do you make that chemistry happen?

Hugo Sarrazin: Yeah, it is not easy when people have spent an enormous amount of time seeped in design thinking or user-centric design, whichever word you want to use. If the person who's applying classic problem-solving methodology is very rigid and mechanical in the way they're doing it, there could be an enormous

amount of tension. If there's not clarity in the role and not clarity in the process, I think having the two together can be, sometimes, problematic.

The second thing that happens often is that the artifacts the two methodologies try to gravitate toward can be different. Classic problem solving often gravitates toward a model; design thinking migrates toward a prototype. Rather than writing a big deck with all my supporting evidence, they'll bring an example, a thing, and that feels different. Then you spend your time differently to achieve those two end products, so that's another source of friction.

Now, I still think it can be an incredibly powerful thing to have the two—if there are the right people with the right mind-set, if there is a team that is explicit about the roles, if we're clear about the kind of outcomes we are attempting to bring forward. There's an enormous amount of collaborativeness and respect.

Simon London: But they have to respect each other's methodology and be prepared to flex, maybe, a little bit, in how this process is going to work.

Hugo Sarrazin: Absolutely.

Simon London: The other area where, it strikes me, there could be a little bit of a different sort of friction is this whole concept of the day-one answer, which is what we were just talking about in classical problem solving. Now, you know that this is probably not going to be your final answer, but that's how you begin to structure the problem. Whereas I would imagine your design thinkers—no, they're going off to do their ethnographic research and get out into the field, potentially for a long time, before they come back with at least an initial hypothesis.

Hugo Sarrazin: That is a great callout, and that's another difference. Designers typically will like to soak into the situation and avoid converging too quickly. There's optionality and exploring different options. There's a strong belief that keeps the solution space wide enough that you can come up with more radical ideas. If there's a large design team or many designers on the team, and you come on Friday and say, "What's our week-one answer?" they're going to struggle. They're not going to be comfortable, naturally, to give that answer. It doesn't mean they don't have an answer; it's just not where they are in their thinking process.

Simon London: I think we are, sadly, out of time for today. But Charles and Hugo, thank you so much.

Charles Conn: It was a pleasure to be here, Simon.

Hugo Sarrazin: It was a pleasure. Thank you.

Simon London: And thanks, as always, to you, our listeners, for tuning into this episode of the *McKinsey Podcast.* If you want to learn more about problem solving, you can find the book, *Bulletproof Problem Solving: The One Skill That Changes Everything* online or order it through your local bookstore. To learn more about McKinsey, you can of course find us at McKinsey.com.

Charles Conn is CEO of Oxford Sciences Innovation and an alumnus of McKinsey's Sydney office. **Hugo Sarrazin** is a senior partner in the Silicon Valley office, where **Simon London,** a member of McKinsey Publishing, is also based.

Designed by Global Editorial Services