

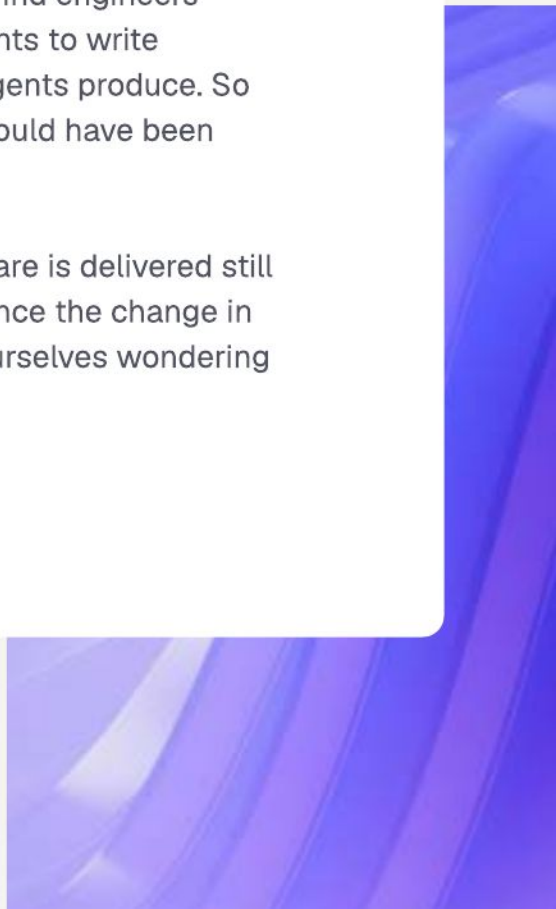
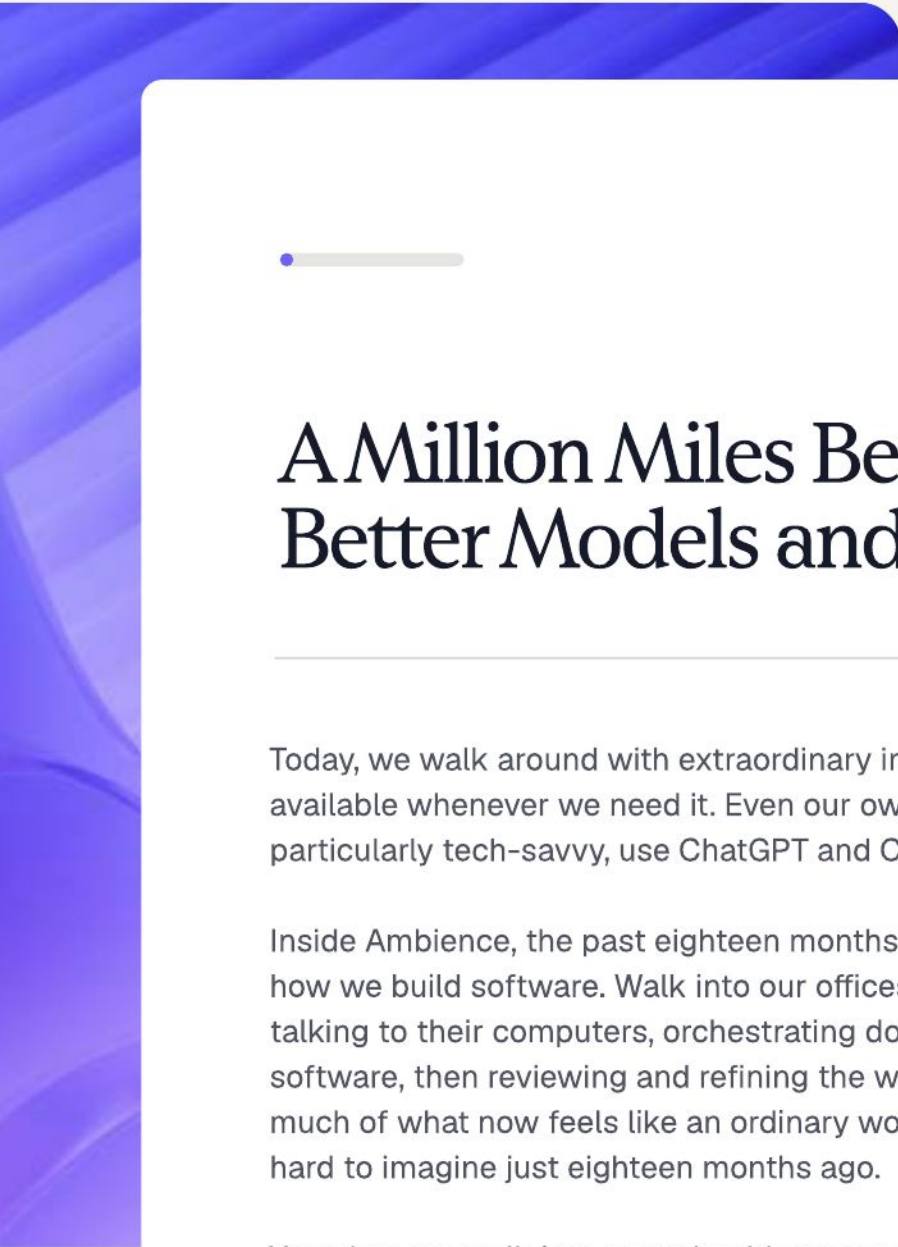


Healthcare Needs Open Infrastructure for AI

Mike Ng

Co-Founder and President





A Million Miles Between Better Models and Better Care

Today, we walk around with extraordinary intelligence on our phones, available whenever we need it. Even our own parents, who are not particularly tech-savvy, use ChatGPT and Claude in their day-to-day lives.

Inside Ambience, the past eighteen months have fundamentally changed how we build software. Walk into our offices and you'll find engineers talking to their computers, orchestrating dozens of agents to write software, then reviewing and refining the work those agents produce. So much of what now feels like an ordinary working day would have been hard to imagine just eighteen months ago.

Yet when we walk into many health systems, the way care is delivered still feels much like it did before this wave of AI. We experience the change in our own lives and throughout our company, then find ourselves wondering where it has gone when we become patients.

You can start a conversation with ChatGPT or Claude on your phone within seconds. But getting an appointment with your doctor still takes weeks or even months. When you finally arrive, you feel like a stranger. You explain your symptoms, your medications, what you've already tried. You answer questions you have answered many times before. After all that waiting, you may have fifteen minutes with your doctor, whose attention is divided between you and the screen. Even when an AI scribe is taking notes, your doctor may still be searching through the chart, piecing together your history and working through tasks in the EHR while trying to listen.

Then you go home, and much of the work of following through falls to you. You chase the follow-up appointment, try to get the referral through, arrange the tests, and make sure someone follows up on the results. If something changes in the meantime, you have to figure out whom to call.

On your own, you may continue to turn to the LLMs. They are available when your care team isn't. But unless your history and care plan are part of that conversation, even a thoughtful answer can miss something that matters in your case. If it conflicts with what your doctor has recommended, you now have another set of questions for the two of you to untangle. ¹

By the time you get back in front of a clinician, there is more to explain, and once again, very little time. You are still the person expected to hold the whole story together. And when you cannot keep up, the gaps in the system become gaps in your care.

At Ambience, we call this the million miles between better models and better care. There is an enormous lag between a new capability becoming available and that capability meaningfully improving a patient's experience, a clinician's work, or the quality of care.

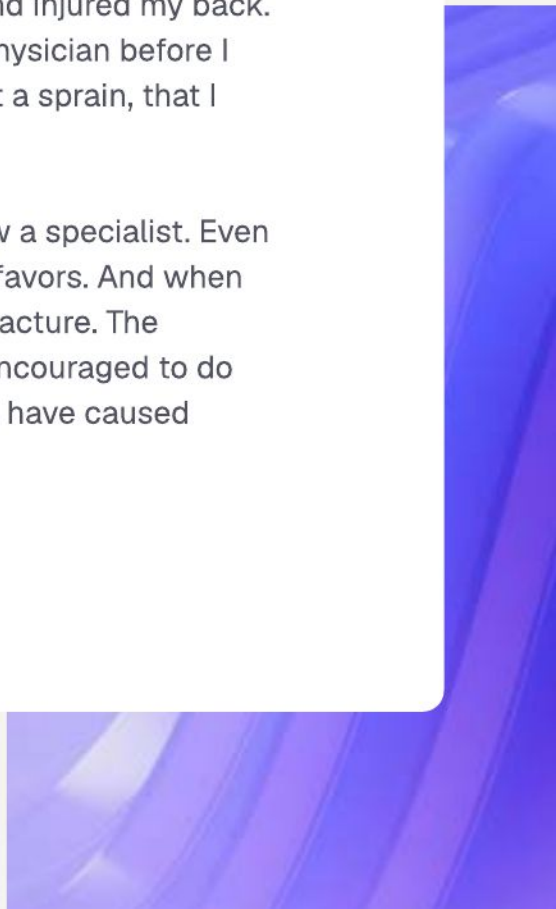
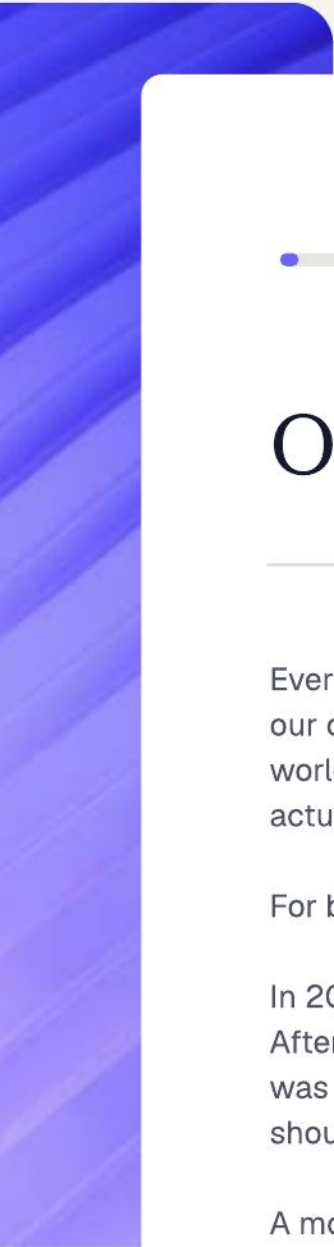
We have come to believe that the ability to close this gap will increasingly determine which organizations thrive. Some will have reorganized their information, tools, and workflows so that increasingly capable models can do more useful work. As the models improve, the value of those structural investments grows. Other organizations may have access to the same intelligence, but still remain constrained by the way they have always operated.

For health systems, this is becoming existential to the mission. An organization that can give clinicians more time with patients, help them make better decisions, and carry their plans through between visits becomes a more compelling place to practice medicine and to receive care. As demand grows and clinicians become harder to find, the ability to deliver more of that care with the people already on the team becomes increasingly consequential.

Closing this gap will require a new class of infrastructure. In the years ahead, AI agents could generate 1,000 times the transaction volume of human users across EHRs and other systems of record. They'll need to reconcile the complex history of the patient across fragmented records. They'll have to deeply understand the caregivers they support and carry their intentions from insight to action at the precise moment care happens. All of this will have to work safely and reliably at scale. These are foundational problems that recur across virtually every AI use case a health system will need to deliver better care, at lower cost, and with a better experience for everyone.

The ecosystem needs an open architecture and shared infrastructure that everyone can build on together: health systems, researchers, and third-party developers alike. Health systems should be able to choose the models and applications that best serve their clinicians and patients, and authorize those tools to use the context and workflows they need under the organization's governance. The teams building those tools should have clear, dependable ways to connect to existing systems and build on shared capabilities.

The question is who will take responsibility for creating this.



Our Why

Ever since Nikhil and I started working together more than a decade ago, our careers have been focused on one question: How do we make the world's best healthcare accessible to every American, at a cost they can actually afford?

For both of us, closing this gap is personal.

In 2012, I ended up in the emergency room after I fell and injured my back. After hours of waiting, I barely got ten minutes with a physician before I was wheeled out of an overrun ER. I was told it was just a sprain, that I should keep moving, and that everything would be fine.

A month later, as things were getting worse, I finally saw a specialist. Even getting that appointment scheduled required calling in favors. And when they redid my imaging, I learned that it was actually a fracture. The specialist informed me that the movement I had been encouraged to do was precisely what I should have been avoiding. I might have caused damage that could not be reversed.

For the next year, here in a country where I had few friends and no family, I had to figure out how to get myself to appointments. I was Googling for basic information: how long I could take my painkillers, when to start physical therapy.

I remember reflecting a lot back then, as I was navigating my care. I decided that if I ever got my health back, I would spend the rest of my career making sure nobody had to endure what I went through ever again.

Today, Nikhil and his better half are navigating a similar journey. Nikhil's partner chose a career as a physician because she wanted to be there for people at their most vulnerable, to give them strength and hope when they needed it most.

But after caring for patients in the inpatient psychiatric ward in San Francisco, she became severely ill with long COVID. In January 2025, she had to leave the practice of medicine. Today she is profoundly disabled, with barely enough energy to even walk to the bathroom on her own.

Yet she and Nikhil have to explain her condition, her symptoms, and her medications over and over again as they move from one appointment to the next.

Between visits, they are arranging testing, fighting for insurance coverage, and even having to help write the prior authorization letters for medications she so desperately needs. Her clinicians are stretched for time. Much of the work of getting the care they recommend falls to her and Nikhil. Despite their privileged access and resources, she is still struggling to get the care she needs.

The national picture makes this reality even more pressing with each passing year. More than 11,000 Americans are aging into Medicare every single day. [2](#) As the population grows older and needs more care, the Association of American Medical Colleges projects that the country could face a shortage of up to 86,000 physicians by 2036. [3](#)

Where a patient lives may add another barrier. A 2019 study of Medicare beneficiaries with complex chronic conditions found that those living in rural communities had a 40 percent higher rate of preventable hospitalization than those in urban areas. Differences in access to specialists accounted for more than half of that gap. [4](#)

And yet, sitting here in 2026, Nikhil and I have good reason to be optimistic.

As early as 2010, Nikhil was doing research at Stanford, where Andrew Ng was chaining together GPUs to train neural networks for image recognition. [5](#) In 2014, while Nikhil was writing his book on deep learning, we embedded with the team that would become OpenAI. Back then, it was a ragtag group of researchers working out of an apartment in San Francisco, experimenting with reinforcement learning and unsupervised learning.

We also saw Pieter Abbeel advance imitation learning, teaching robots to learn from human demonstrations. And we got to learn from Jeff Dean, who later became Google's Chief Scientist and was an angel investor in both of our healthcare AI companies. His work helped scale deep learning through Google Brain, whose researchers went on to help develop the Transformer architecture. [6](#)

In 2018, one of our housemates was working on a problem that had frustrated AI researchers for years: getting a model to write at a human level. Some of the smartest people we knew had spent years on it. Nikhil and I had struggled with it ourselves. A model might produce a plausible sentence, but ask it to hold a thought together through a paragraph, let alone an essay, and it would lose the thread.

And then one night, our housemate showed us a preview of GPT-2 over dinner. They could barely contain their excitement. The prompt was to write a short science fiction romance about two robots—the sort of task that would have made our earlier models fall apart. It took a couple of attempts to get a good result, but it finally produced a story that held together and read as though a person had written it.

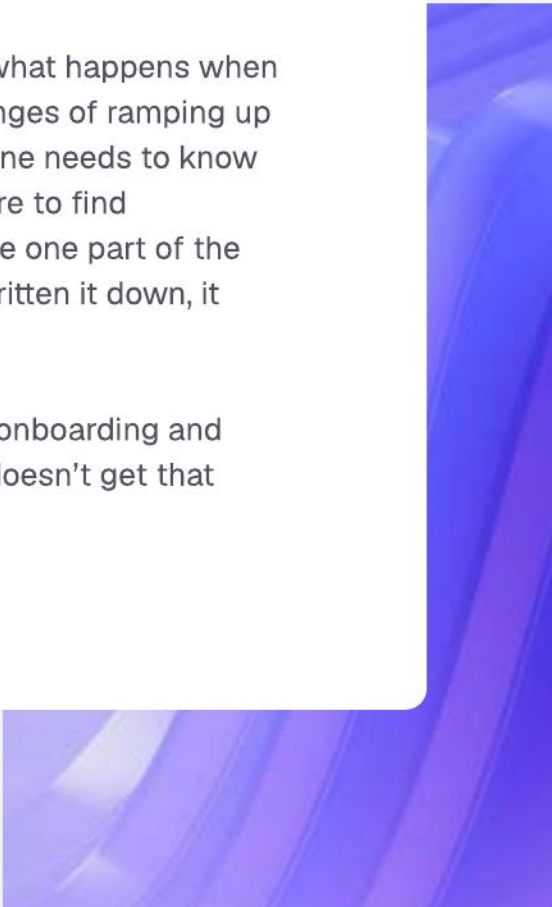
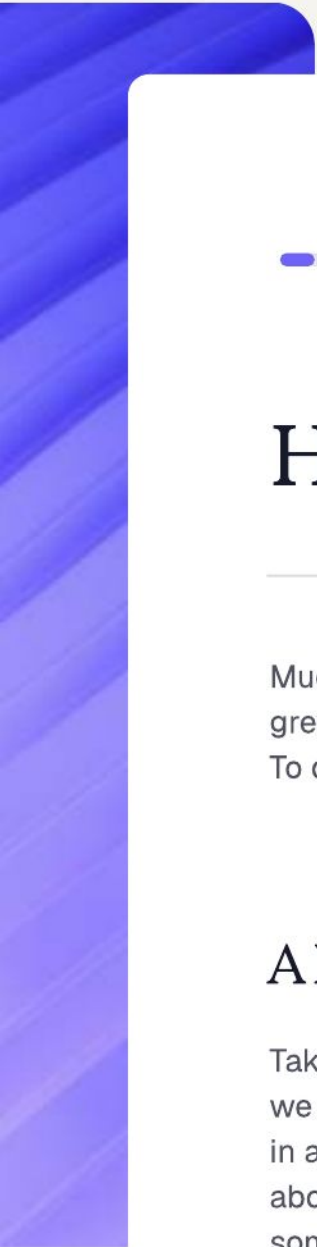
What was remarkable was how it had learned to do this: by predicting the next word across millions of documents downloaded from the internet. An elegantly simple learning mechanism running on an elegantly simple architecture, which is what made it possible to scale up for the first time ever. [7](#)

In 2019, our friends at OpenAI began running experiments to find out how far they could push this capability. They were increasing model size, adding data and compute, and finding that improvements in language prediction followed a consistent pattern. [8](#) It was becoming possible to estimate how well a larger model would predict text before training it. We were starting to have a basis for anticipating what another few years of progress could bring.

When we started Ambience in 2020, we felt we were at the beginning of an exponential. In a few years' time, these models could become so capable that they would forever change the way we work.

In the six years we've spent building Ambience, the exponential in AI that we anticipated has largely taken hold. ChatGPT launched on November 30, 2022. Within two months, it had an estimated 100 million monthly active users. [9](#)

In the process of building Ambience, Nikhil and I have sharpened our answer to the question that has guided our careers: how do we make the world's best healthcare accessible to every American, at a cost they can actually afford? Put simply, it's to build the open technology infrastructure and become the care transformation partner that health systems need to close the million miles between better models and better care.



Harnessing the Exponential

Much of our thinking about how to help health systems close this gap grew out of our own experience trying to put AI to work inside Ambience. To do that, we had to fundamentally rethink how the company worked.

A New Class of Infrastructure

Take our engineering organization, as an example, and what happens when we onboard a new employee. One of the hardest challenges of ramping up in a new job as an engineer is that much of what someone needs to know about our software lives in other engineers' heads: where to find something, what it was supposed to do, and when to use one part of the system rather than another. Even when someone had written it down, it wasn't always easy to find.

A new engineer could learn by asking questions during onboarding and tracking down colleagues when they got stuck. But AI doesn't get that same privilege. At least, not as easily.

And so we had to make that knowledge available to AI much more deliberately. We documented what different parts of our software were for and the reasoning behind how they were built. We organized the codebase and its documentation so agents could find what they needed. We made our development practices explicit, including how we test products and how we build, train, and improve our models.

This work extended across the company, as we started to transform each department. We rethought how we stored and indexed information, and how we documented our thinking as we made decisions: what we were trying to solve, which alternatives we had considered, and why we chose the path we did. People and agents could then inherit that understanding and build on it in their own work.

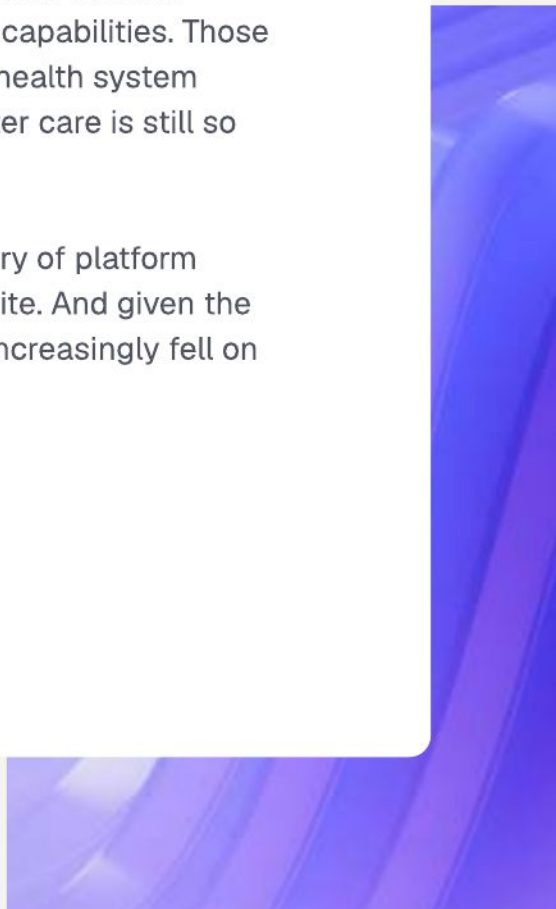
Then there were the tools. We wanted an agent to be able to write code and submit it for review, launch an environment in which to test it, or draft an email or document. Sometimes the useful next step was to put an insight in front of the right colleague in Slack, while there was still time to act on it. Each of these possibilities required a connection to the systems where our work happened, and a way for the agent to carry out the task.

Every connection also raised questions about authority and security. We had to decide what an agent should be allowed to read or change, which actions required a person's approval, and which should be blocked altogether. We needed environments that constrained what agents could do, records that let us see what they had done, and ways to intervene when something went wrong. Those boundaries had to be enforced by the systems around the model, independently of its judgment.

Taken together, this was an entire layer of infrastructure we had to build inside Ambience to turn intelligence into useful work. It changed what we could do when a better model arrived. The model could draw on our context, use our tools, and work within the boundaries we had established. We could put its new capabilities to work sooner because so much of what it needed was already in place.

This is not a new lesson. We saw a similar pattern in the history of the personal computing industry. The race for faster CPUs drew enormous attention, but processor speed alone could take users only so far. An entire industry developed to supply the surrounding infrastructure—memory, storage, interfaces, operating systems, and applications—that made computing power useful. Companies such as Microsoft and Apple built their businesses around that opportunity. By the end of 2010, those two companies alone had a combined market capitalization more than four times that of Intel and AMD. [10](#)

The AI megatrend in software engineering is already showing us this lesson in real time. Across technology companies, including Ambience, much of the practical value of the past eighteen months has come through innovations such as Claude Code, which gave the agent access to a codebase and tools, managed context as it worked, and enabled it to make changes, test the results, and revise its approach. [11](#) Smarter models made these products possible, but it was the infrastructure around those models—what the AI community refers to as an “agentic harness”—that made them useful in the day-to-day work of building software.



The Analog for Health Systems

This brings us to an important question: who is closing this infrastructure gap for health systems? Who is building the foundations required to turn AI into a productive member of every care team?

Historically, the first place a health system might look is to its EHR vendor. But in our work with health systems, we encountered profound infrastructural deficiencies. We collectively realized that our current systems of record lack entire categories of critical core capabilities. Those deficiencies are in large part why, when we walk into a health system today, the million miles between better models and better care is still so profoundly felt.

The core challenge is that this is an entirely new category of platform technology. It requires a fundamental architectural rewrite. And given the magnitude, the onus of closing this infrastructure gap increasingly fell on Ambience.

When we first started working with the country's leading health systems, we only had our singular flagship use case: the Ambience AI scribe. But the sheer speed and scale of Ambience's impact, even relative to the other solutions in our category, shocked our partners. Within a couple of months, we would see the vast majority of care teams using Ambience's scribe for almost every single visit. Very quickly, clinicians refused to ever practice again without it.

As this transformation took shape within their walls, our partners were motivated to do more with us. Within a year, we went from one use case to working on more than a dozen. Our customers were pulling these new capabilities out of our hands.

Over the past two years, this has become a virtuous cycle. As we developed a deeper understanding of our clinicians, their patients, and the organizations in which they practiced, we could bring forward insights that were more useful in their daily work. As we built the tools to act on those insights, we could take more work off their plates. Clinicians came to rely on Ambience for more of what they did, giving us further opportunities to understand their work and earn their trust to do more on their behalf.

This is why adoption matters so much to the future we are building with our partners. When the vast majority of care teams use Ambience for almost every visit, each new capability can reach a substantial share of the care an organization delivers. We have a place in the workflows where decisions are made, an understanding of what the care team is trying to accomplish, and a relationship with clinicians who increasingly want our help carrying those intentions forward. What began with writing the note has become an opportunity to extend the reach of the care team.

Earning that broader role required us to solve a growing set of infrastructure problems. As we expanded our product portfolio, we kept encountering the same technically challenging obstacles. Different clinical applications needed many of the same foundational capabilities, and those capabilities were missing from the systems of record we were building on.

We tried, alongside our health system partners, to get the incumbent vendors to close these gaps. Time and again, we came away without a clear answer about how they would be solved, much less when. Once our partners had already seen what AI could do for their clinicians, they were not prepared to put their roadmap on hold.

To deliver on these outcomes, we took on the work of building this missing infrastructure. Over time, we came to a profound realization: the infrastructure we had created to make intelligence useful inside our partners' clinical environments was quickly becoming the most important thing we had ever built. Yes, our AI scribe had changed lives, helping clinicians who were otherwise going to retire or leave medicine rediscover joy in practice and choose to stay. But looking ahead to the next five years, we and our partners started to recognize that the infrastructure beneath Ambience's scribe, and the growing array of products that came after it, could have an order of magnitude greater impact on the quality of the care they could deliver, the cost of that care, and the collective experience of their patients and clinicians. Without it, there was no credible path to infusing every care team with more AI, extending their capacity, and ultimately turning the industry's vision for AI-enabled care delivery from a concept into reality.

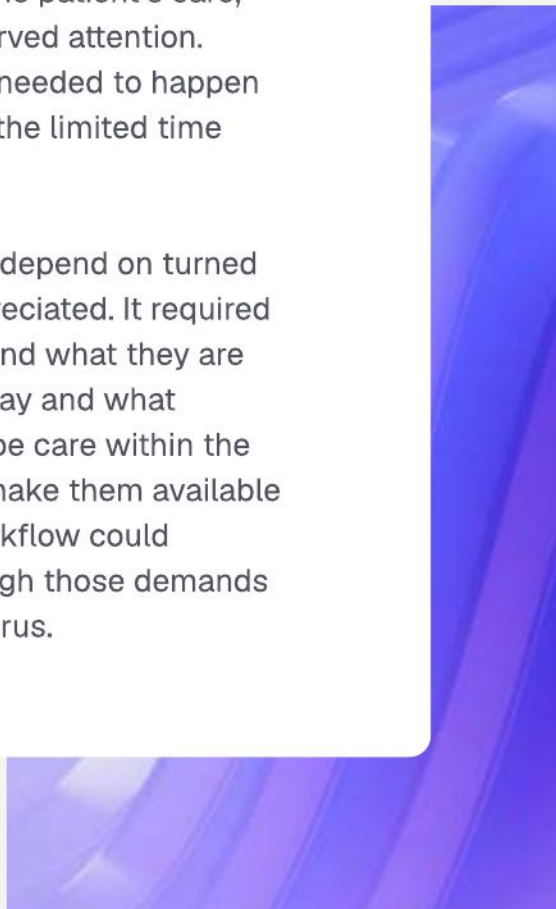
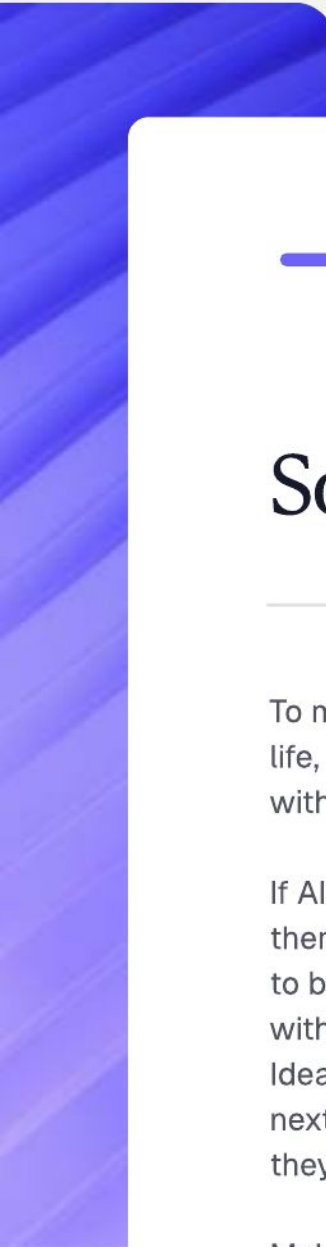
As a result, in 2024, we began to reorient the full force of Ambience to accelerate this core infrastructure work. Today, we're finally excited to announce this new class of clinical AI infrastructure for health systems, **Ambience Chorus**, which now powers Ambience deployments at our health system partners.



Chorus brings together *a System of Context, a System of Action, and a System of Governance*. Together, they give AI an understanding of the people it serves, the tools to work on their behalf, and the safeguards and measurement needed to oversee that work and assess its impact.

Chorus already provides these foundations for Ambience's applications. Opening these foundations to the broader ecosystem would allow health systems, researchers, and third-party developers to build their own applications on the same infrastructure.

An application built on Chorus should be able to draw on the patient, clinician, and organizational context it maintains, use its integrations to carry out work across clinical systems, and operate within the permissions and safeguards established by the health system. A team developing a new capability for a particular specialty or care pathway could then begin with those foundations already in place, concentrating its effort on the clinical problem it wants to solve.



Solving for Context

To motivate some of the challenges behind bringing this infrastructure to life, let's start with one of the first use cases clinicians asked us to help with after the scribe: preparing for a visit.

If AI could help them write their notes, could that same intelligence help them get ready to see the next patient? The clinicians we served wanted to be able to walk into the room already caught up on the patient's care, with a clear sense of what had changed and what deserved attention. Ideally, the system could think through everything that needed to happen next and craft the perfect agenda to make the most of the limited time they would have in the room.

Making that preparation useful enough for clinicians to depend on turned out to be far more demanding than we had initially appreciated. It required three kinds of understanding: how the clinician works and what they are trying to accomplish; what is true about the patient today and what matters to them; and what resources and priorities shape care within the organization. We needed to keep all three current and make them available across applications, so that what we learned in one workflow could improve the help we provided in another. Working through those demands helped shape the *System of Context* at the heart of Chorus.

Understanding the Clinician

The first challenge was giving AI a deeper understanding of the clinician we were trying to augment. Perhaps a relatable analogy for many executives is the difference between working with an exceptional executive assistant and a mediocre one. With a mediocre assistant, you may hand off the task while still carrying much of the mental work required to get it done. You keep supplying the background, deciding what needs attention, and correcting work before you can use it. You can end up spending so much time managing the assistant that they add more cognitive load than they remove.

On the other hand, working with an exceptional executive assistant can feel almost magical. They understand what you're responsible for, what you're trying to accomplish, and what needs your attention. They know when you want the full background and when you need someone to get to the point. Over time, they learn to anticipate what you need, and you spend less of your day explaining yourself.

We wanted clinicians to have that same experience with Ambience: an assistant that understood their work and preferences deeply enough to anticipate what would help.

The Complexity of Clinician Context

Doing that meant appreciating and accounting for the vast diversity of medicine practiced across the specialties and subspecialties at our partner health systems. Consider a geriatrician seeing a patient with dementia and two dozen chronic conditions. Their responsibility is to hold the patient's care together across those competing needs: understand how the problems affect one another, distinguish what is well managed from what is getting worse, and decide where attention is most urgently needed. A neuro-oncologist may be approaching a particular disease at extraordinary depth, tracing the course of treatment and the evidence of its effects to determine whether to continue on the current pathway or change direction.

One is orchestrating care across many problems; the other is bringing highly specialized judgment to bear on a particular one. Preparing them well requires understanding these different jobs and finding the information that supports each of them.

The resulting experiences can feel like entirely different products. The assistant draws on different context, brings forward different information, and dynamically shapes the interface around the decisions each clinician needs to make. Understanding the work changes the experience of using the software.

The work changes again with the purpose of the visit. For a new consultation, preparation establishes the background, but much of the note depends on what the clinician has yet to learn during the encounter. In an established oncology follow-up, much of what needs to be documented is already known before the patient arrives and is not verbalized during the visit. In some of these visits, as much as 90% of the note could be drafted in advance, with the encounter supplying the new findings and decisions needed to complete it. Understanding which kind of work lies ahead changes how much the assistant can do before the visit begins. Preparation can become the beginning of documentation.

Alongside that understanding of the clinical work, we also needed to understand the individual. Two physicians in the same specialty can have very different preferences. How a clinician writes can reveal both how they express themselves and how they organize their clinical thinking. That understanding should shape what we put in front of them as much as what we write on their behalf.

Those preferences also depend on the situation. A physician who wants a terse inpatient update may want a much richer longitudinal history before a complex new consultation. The same person may explain things to a patient quite differently from how they communicate with a specialist colleague. We need to learn how they work in different circumstances, so that an insight from one setting can help us in another without being applied indiscriminately.

Encoding Clinical Taste

It is easy to assume that smarter models would already know what excellent preparation looks like for a given specialty and type of visit. Yet even as models have become dramatically more capable, we have continued to encounter a gap between medical knowledge and clinical taste. A model may know a great deal about every condition in the chart and still fail to recognize which information this clinician needs for the decision in front of them, what deserves emphasis, and what can be left out. Even when a summary contains the right information, its usefulness depends on how that information is organized and presented. A timeline can make a treatment history easier to follow; a visualization can make a trend apparent across dozens of individual results. These choices require their own understanding of clinical work: what a clinician needs to see at a glance, and what supporting detail should be available when they need to look more closely. The challenge for frontier labs is that they are far removed from day-to-day clinical practice. Without a deep understanding of the work, it is difficult to define what quality looks like, how to measure it, and how to continuously improve performance.

Closing that gap required us to build a team with an unusual combination of skills. We brought clinicians with software engineering backgrounds into close collaboration with leading AI researchers. Together, they had to articulate what excellent preparation looked like, then build a system that could express the nuances of clinical work in software. That meant accounting for differences across specialties, visit types, and individual clinicians' preferences. Making that clinical taste available through a dedicated clinical harness around the model substantially improved clinician engagement.

Continuous Learning as Infrastructure

The value of that understanding extends well beyond preparation. It should follow the physician through the rest of their work, including between outpatient and inpatient care, while accounting for how their responsibilities and preferences change. Reconstructing all of this from scratch for every new use case would be an enormous undertaking and create unnecessary work for clinicians, who would need to teach each new assistant how to help them. A shared *System of Context* lets each application begin with what we have already learned.

Each application also gives us an opportunity to learn something further about the clinician. Documentation might teach us how they organize their reasoning; preparation might teach us what they want brought to their attention. Those learnings should return to the *System of Context*, together with the circumstances in which they apply, so that what we learn in one application can improve the help the clinician receives from another.

With this *continuous learning* loop built into the clinical harness, a new application can inherit years of understanding from the products that came before it. It still needs to learn what is particular to its own task, but what it subsequently learns can improve the applications the clinician already relies on. The whole system can become more useful as each individual application gets used.

Understanding the Patient

Preparing the clinician also meant understanding the patient they were about to see. At first glance, this just seemed like a matter of giving AI access to the chart. Years of notes, diagnoses, medications, and test results were already there; surely a sufficiently capable model could pull them together and tell the clinician what they needed to know before the visit.

In Search of a Durable Source of Truth

One of our earliest discoveries was that a model could ingest the entire chart and still give us a misleading picture of the patient.

Imagine a patient whose cardiology notes describe a mechanical mitral valve and long-term warfarin to prevent dangerous blood clots. The outpatient medication list still shows warfarin as active. A recent hospital note, however, says it was held during a serious gastrointestinal bleed, with a plan to reassess anticoagulation after the bleeding was controlled. The record available to the assistant does not confirm whether that reassessment happened or what the patient is taking now.

The harsh reality is that, despite our best intentions, the record is often closer to a filing cabinet of observations than a coherent account of what is true about the patient today.

Keeping that record coherent is a job in its own right. It requires the equivalent of a librarian whose ongoing responsibility is to maintain its integrity: reconciling new observations with what came before, distinguishing active problems from those that have resolved, and making sure medication changes are reflected in our understanding of the patient. Today, clinicians do pieces of this work in the limited time they have. Much remains unresolved, leaving each successive care team to ask many of the same questions and reconstruct what is actually true.

On the surface, this can look like a routine data-cleanup problem: find the latest information and update the medication list. But in this patient's case, the older cardiology note still contains an essential fact. The mechanical valve creates an ongoing need for protection against clots; the hospital note explains why warfarin was interrupted. The system has to interpret those facts together, look for evidence of a restart or a temporary alternative, and recognize what remains unknown. That requires sophisticated clinical reasoning about the course of care.

Bad context can be worse than no context. If an assistant carries the active medication list into instructions to continue warfarin, it could encourage resumption while bleeding remains a concern. If it treats the hospital's interruption as a permanent discontinuation, it could obscure the need to address anticoagulation, leaving a high-risk patient without protection against valve thrombosis or stroke. An apparently cleaner account of the patient's medications could conceal the very issue the care team needs to resolve. [12](#)

Better models can improve how we interpret the evidence in the chart. But if the available record never establishes whether this patient restarted warfarin, a larger context window or more sophisticated reasoning cannot supply that missing evidence.

We needed a system that could reconcile contradictions on its own when the clinical evidence was sufficient, recognize when it needed a clinician's judgment, and turn missing information into a specific question for the patient or care team. Once that question was resolved, the answer needed to return to the shared *System of Context*, along with its source and the circumstances in which it applied. The next application or care team could then inherit what had been learned, instead of having to resolve the same contradiction from scratch.

Understanding the Person Beyond the Chart

Preparing a clinician for a visit means catching them up on what is happening clinically. It also means bringing forward what matters to the patient and what might make following through on care difficult.

Historically, the electronic medical record has centered on clinical observations, treatment decisions, and documentation for billing. Yet a great deal of effective care depends on connecting with a person well enough to help them make changes that are difficult to sustain. The best clinicians develop a deep understanding of what matters to their patients, what motivates them, and what stands in their way. That understanding shapes how they approach conversations that might otherwise end with a recommendation the patient struggles to act on.

A patient may have heard advice about exercise many times without changing their routine. Learning how much it means to them to attend their granddaughter's graduation in two years can give the clinician a different starting point: connecting the changes they are recommending to something the patient wants for their own life. The clinician also needs to understand the circumstances that determine what is possible. Whether the patient can afford their medications, get to appointments, or find help at home can matter as much as their willingness to follow a plan.

This understanding has rarely received the same attention as the clinical information we document. It may emerge in a conversation, be buried in a note, or stay in the memory of one clinician. There is often no clear place to maintain it as the patient's circumstances and priorities change, or a reliable way for another member of the care team to inherit it. A clinician may have spent years learning how to connect with a patient, yet the next person helping them may have to start from the beginning.

There is a second challenge: even when important context has been captured, it may live somewhere the assistant cannot see. A population health platform may hold the patient's accountable care organization (ACO) attribution and identified gaps in care. An outside specialist's assessment may arrive by fax and sit in a separate document store.

A call-center conversation stored in a CRM may explain why the patient has been unable to schedule a test. Each can change what the care team needs to know or do next. An assistant working only from the EHR could miss information that the organization already has and the patient has already supplied.

This is why the scope of the *System of Context* needs to extend well beyond the EHR. It needs to bring together relevant information from other systems of record and become a durable home for understanding that those systems have not consistently maintained. A clinician preparing for a visit should be able to build on what others have learned about the person: what matters to them, what has helped them make progress, and what has stood in their way. A future agent helping that patient manage a chronic condition would need the same understanding to offer support that fits the realities of their life.

Engineering Context at Scale

As we started to expand our pre-charting capability from a small pilot cohort to the broader enterprise, the scale of the engineering problem became clear. At one of our leading academic medical center deployments, our initial pre-charting product generated more transaction volume in the health system's EHR than all other human and application activity combined. This was one use case, rolled out to roughly 10% of their clinicians. If preparing a tenth of their clinicians for visits could create that much load, what would happen as we introduced more applications and agents across the health system?

We had to fundamentally rethink the architecture for retrieving, preparing, and maintaining patient context. It needed to support a health system's full patient population and a growing number of applications, without overwhelming the systems its clinicians depended on.

That required us to:

- Retrieve longitudinal histories in bulk and prepare context on our own infrastructure, substantially reducing the work imposed on the EHR.
- Keep that context current by incorporating event feeds and selectively retrieving live data when freshness could change a clinical decision.
- Reconstruct a faithful history when results arrive late, documents are revised, retrieval paths return different versions of the same information, or one episode of care spans multiple encounter records.
- Predict retrieval load and distribute it across a 24-hour window, accounting for the densest patient charts and reserving capacity for information needed during care.

We also had to coordinate that work across applications. If five applications need the same information, we should retrieve and process it once, then make the result available to all five. As new information arrives, we update that shared understanding. Without this coordination, dozens of agents independently querying the same records can effectively become a self-inflicted denial-of-service attack on the health system. A shared *System of Context* lets us manage that demand collectively.

Model consumption added another constraint. At a major academic medical center, a patient's longitudinal record can contain millions of tokens of data, with images and faxed documents adding further processing demands. Sending that entire history to a model for every task can generate an enormous bill.

These problems could not be solved independently. Preparing more context in advance reduced pressure on the EHR, but required us to account for everything that changed afterward. Being more selective about what we gave the model reduced costs, but increased the risk of leaving out something clinically consequential. We had to preserve the information the care team needed, keep it current, and make it affordable to use across the full patient population—all without overwhelming the systems on which care depended.

When EHR vendors pass through token consumption costs, the health system bears the expense of those engineering decisions. Under our performance-based contracts, managing those costs is our responsibility. If we consume more than we should, that is ours to absorb. A health system should not have to subsidize inefficient engineering to get the clinical value it was promised.

Building scalable patient context took more than 40% of Ambience's engineering resources for almost two years.

A successful pilot can give a health system every reason to expand without revealing how much infrastructure that expansion will require. These problems become unavoidable when more clinicians begin using the product, more applications come online, and AI starts becoming part of everyday care. By then, the organization has already built expectations around what the technology will deliver. Whether it can fulfill those expectations depends on whether its partners have solved the infrastructure problems that success itself brings into view.

Understanding the Organization

There is a third kind of context that is easy to overlook when we think about preparing a clinician for a visit: how care actually works inside their organization. An assistant can understand the patient and the clinician yet know very little about the health system's specialists, clinical programs, and support services, or the organization's priorities for care.

What the Organization Can Offer

Health systems continually invest in people, programs, and infrastructure to improve the care they can deliver. The success of those investments depends in part on whether frontline teams know what is available, recognize when it could help, and know how to use it. In a large organization, that understanding does not reach every care team simply because a program has launched.

An assistant that understands those resources could help close that gap by bringing them to the care team's attention when they are relevant to a patient's needs.

Consider a health system that has recruited leading specialists and built a new movement-disorders program. Elsewhere in the same organization, a physician may still be referring patients along familiar pathways, unaware of what the new program can offer. A patient who could benefit from that expertise may end up seeking care outside the system, while the team the organization worked so hard to assemble remains underused. For leaders wrestling with referral leakage, this is one way an investment in clinical excellence can fail to reach the patients it was intended to serve.

The assistant needs to understand which patients a program can help, where its services are delivered, and how to access them. Where the appropriate expertise lies outside the organization, it needs to know the established referral relationships too. That understanding can bring a relevant option into the conversation while the clinician is still deciding what to recommend

The same applies to resources that may be much less visible. An organization might partner with a program that rewards eligible low-income patients for taking their prescribed medications, offering gift cards or transportation vouchers. A physician preparing to see a patient with hypertension may have no idea the program exists. An assistant that knows about it could bring forward the eligibility criteria and enrollment pathway before the visit, creating an opportunity to discuss a resource the organization has already invested in making available.

Some of this knowledge lives in the EHR, but much of it is scattered across directories, announcements, spreadsheets, and people's heads. It changes as specialists join, programs expand, and referral pathways evolve. The *System of Context* needs to maintain that understanding so that the organization's capabilities become accessible to the applications supporting its care teams.

Bringing Organizational Priorities to the Point of Care

Understanding the institution also means understanding the changes it is trying to make. Quality initiatives, nursing policies, and clinical pathways reflect decisions about how care should be delivered. Payer commitments and the strategy around assuming financial risk can further shape which improvements the organization prioritizes.

But context especially matters because the precise nudge that's indicated depends on what exactly an organization is trying to improve and where its care processes are currently breaking down. A generic reminder can both be clinically sensible and yet still miss the local problem entirely.

For example, a health system working to improve its performance on the SEP-1 sepsis measure might discover that clinicians are recognizing the need for treatment and placing orders, but administration is delayed. A generic prompt to recognize sepsis earlier would target a step clinicians are already completing. The assistant needs the quality team's findings and the agreed improvement plan to understand which part of the process needs attention, who is involved, and when a nudge would be useful.

Those findings and decisions are organizational knowledge. The work done in a chief quality officer's office should be able to inform the assistance a care team receives on the floor. Making that understanding available to clinical applications creates an opportunity to bring the organization's improvement efforts into the daily work where they need to take effect.

AI-assisted analysis could also help organizations develop new understanding—for example, by surfacing patterns that quality teams had not previously recognized. Once reviewed, those findings and the resulting improvement plans could become part of the *System of Context* available to applications supporting care delivery. What the organization learns would then change the help its care teams receive.

From Intelligence to Useful Insight

The preparation we wanted to make possible draws on all three kinds of understanding. It is shaped around the decisions this clinician needs to make, grounded in the patient's current situation and priorities, and informed by what the organization can offer and what it is working to improve. The result should be an agenda that brings the right issues and opportunities into the room while there is time to address them.

Those requirements recur across the care team's work. A referral letter should reflect the referring clinician's reasoning, bring forward the patient information the recipient needs, and account for how care is coordinated locally. A future agent calling a patient or family on the care team's behalf would need to understand the patient's circumstances, the clinician's intentions and communication preferences, and the resources the organization can offer to help them follow through.

We are building *Chorus's System of Context* so that applications built by Ambience, health systems, researchers, and other developers can draw on and contribute to this shared understanding. A team building a laboratory coordination agent, for example, should be able to access the relevant care plan, the patient's circumstances, and the organization's available services through interfaces governed by the health system. When the agent learns that transportation is preventing the patient from completing a test, it could contribute that information, together with its source, for incorporation into the context available to the care team and other authorized applications.

The work of retrieving, reconciling, and maintaining that context at scale would become a shared foundation for the ecosystem. Developers could concentrate on the particular clinical problem their application is intended to solve. Health systems could adopt new applications and more advanced AI capabilities as they emerge while continuing to build on the understanding they have already developed of their clinicians, patients, and organization.

Solving for Action

Even when a care team knows exactly what should happen next, a great deal of work stands between that decision and the patient receiving care. Chorus's *System of Action* is the infrastructure we are building to help carry that work through—within the encounter, across clinical workflows, and beyond the four walls of the clinic.

From Clinical Intent to Completed Work

The *System of Action* starts with advancing work inside of the encounter. Consider a clinician who has diagnosed type 2 diabetes and discussed a plan to start metformin and follow up in four weeks. An AI scribe may capture that conversation perfectly. But the clinician still has to turn back to the EHR: search for the medication and specify the prescription, select a diagnosis from the local catalog, and make sure the assessment, plan, and patient instructions reflect what they have just agreed with the patient. The decision has been made and explained, but the clinician is still moving between screens to put it into effect.

Getting an agent to take over that work meant solving two problems that a better model alone could not resolve. We needed ways for it to perform the required operations in the EHR, and we needed it to understand the local rules that governed each operation.

Over the last several years, we have worked closely with our health system partners to gain access to the universe of private EHR APIs that are notoriously difficult for third-party vendors to access. But access to an API did not mean the same action would work the same way at every hospital.

Write-back operations have to account for how each organization represents diagnoses, medications, and procedures, or configures workflows such as scheduling. Take diagnosis write-back, for example: the selected diagnosis has to map to an entry the institution's EHR will accept. Whether a particular term can be written back depends on the institution's local catalog and rules: which entries are available and active, and what level of specificity is accepted. A model could correctly identify "type 2 diabetes without complications" and still fail to add it to the encounter because it had selected a term that the local system would not accept. To make this work, we had to build a translation layer that could ingest each institution's catalogs and rules and resolve the local mappings our applications needed to carry out these actions. Making this translation layer available to other developers would spare them from rebuilding the same local integrations and configurations for each new application.

Even after years of work expanding our EHR integrations, we still encountered actions for which no API was available. For these tasks, we began developing agents that could operate the interface itself. Our dictation work gave us a starting point: integrations that let us identify the field a clinician was working in, read what was already there, and insert text at the cursor. We are building on that foundation with computer-use capabilities that can interpret the screen, navigate the interface, and enter information. These capabilities are now beginning to go live at several of our health system partners.

Consider, for example, how Ambience's new computer-use capabilities help automate workflows for the clinician at the end of the visit.

An agent navigates to patient instructions and inserts the prepared after-visit summary. It recognizes that the encounter is missing a CPT code, determines and enters the appropriate code, submits the associated charge, and carries the encounter through closeout—all while keeping the clinician in the driver's seat.

This is how the *System of Action* enables more of the work inside an encounter to be automated. The clinician explains and directs the care, reviews what requires their judgment, and has an assistant carry out the corresponding work in the software. The EHR remains where that work is recorded and coordinated, but the clinician no longer has to operate it personally at every step.

We welcome the administration's efforts to remove unnecessary barriers to putting capabilities like these to work for clinicians and patients. ONC's December 2025 guidance clarified that interference with automated access to and use of electronic health information, including through agentic AI, can implicate information-blocking rules. [13](#) The proposed HTI-5 rule would make the inclusion of automated access and use more explicit and tighten the conditions for certain exceptions. [14](#) These are important steps toward giving health systems greater freedom to choose the tools that best serve their clinicians and patients.

Coordinating Work Across the Care Team

But the work of caring for a patient often continues beyond the scope of an individual encounter. An insight or decision from one clinician's workflow may require another member of the care team to follow through hours, days, or even weeks later. For example, in the inpatient setting, a hospitalist may identify an unresolved issue that needs to become a task for a nurse on the shared team task list. In the ambulatory setting, a psychiatrist's management plan may guide a primary care physician over subsequent visits.

We encountered this early in our work with behavioral health and psychiatry service lines, where there were simply not enough specialists to meet patient demand for care. Psychiatrists were trying to help their primary care colleagues manage more of these patients by writing detailed plans. But creating those plans took considerable time, and even a comprehensive document could leave the receiving physician uncertain about what they could confidently manage themselves and when they needed the psychiatrist's help.

This is where Ambience's *System of Action* was able to make a major impact. Ambience captured the consultation, helped the psychiatrist compose a management plan, and prompted them to clarify decisions that might otherwise have been left implicit: what primary care should monitor, which treatment adjustments they could make, and what changes should bring the patient back into specialty care.

One recurring use case was helping psychiatrists support primary care physicians in managing adults taking lithium for bipolar disorder. Lithium can be highly effective, but managing it requires careful follow-up. Alongside assessing symptoms and side effects, clinicians use blood tests to monitor the amount of lithium in the bloodstream and check kidney function. The plan needs to make clear how those findings should inform treatment, including when a result calls for further investigation rather than an immediate change in dose. [15](#)

What made this work powerful was that the psychiatrist's plan became an active part of the software supporting the primary care clinician. AI helped turn their intentions into something that behaved like a small program: a patient-specific module whose instructions and conditions shaped how Ambience assisted the next member of the care team.

Before a primary care visit, that module helped Ambience interpret the patient's latest information in the context of the psychiatrist's plan and identify what needed attention. During the visit, when the physician asked a question through Ambience's chat capability, the answer drew on the specialist's specific intentions for that patient, alongside the chart and relevant guidelines. If a lithium blood level came back low, for example, the psychiatrist's instructions could prompt the physician to check whether the patient had missed doses and when the blood sample was taken before considering a dose increase. [16](#)

The plan also established when the psychiatrist needed to become involved again. Suppose repeat blood tests showed a meaningful decline in kidney function. Because the kidneys remove lithium from the body, that change could make a previously appropriate dose unsafe. The patient might need closer monitoring, a different dose, or reconsideration of the medication itself. ¹⁷ If the change met the psychiatrist's criteria for reassessment, the module could bring that need into the primary care clinician's preparation and decision support, together with the results and intervening history that explained why specialist input was needed.

For the clinicians involved, this changed the confidence with which they could share responsibility. Psychiatrists felt more comfortable having appropriate patients managed in primary care, and primary care physicians felt better equipped to do right by those patients. This is a practical way to extend scarce specialist capacity: support more ongoing management in primary care, preserve a clear path back when treatment requires specialist judgment, and make more of the psychiatrist's time available to patients whose needs cannot be managed elsewhere.

That experience helped us see a broader role for the *System of Action*: helping a care team carry a plan through together, even when the work spans multiple clinicians and visits. Each person needs to understand their part, have the support to carry it out, and know when to bring someone else in. Getting that coordination right is what connects the decisions made during an encounter to the care a patient actually receives.

Beyond the Four Walls of the Clinic

But coordinating the care team's work across episodic encounters is only part of the challenge. Most of the time, patients are living their lives at home. They are picking up and starting medications, navigating their treatment plan, and monitoring symptoms—often without a clinician alongside them to help. Care teams want to be more proactive and involved during these intervals, but their time, attention, and expertise are already stretched thin.

As Ambience has taken more work off clinicians' plates, they have increasingly asked us to extend that help beyond the encounter. Could pre-visit preparation include reaching out to the patient and asking some of the routine questions that would otherwise take up time during an annual wellness visit? When a clinician is worried about how a patient will fare on their own, could an agent follow up after the visit to help them carry out the care plan—checking whether they picked up their medication, understood the instructions, and were responding as expected? Could that same assistant help them overcome obstacles to following through, and bring the relevant context back to the care team when something needed their attention?

Those requests reflect the trust we've earned. We're now working with our health system partners to build agents that extend the care team's reach beyond the exam room.

The power of this capability becomes especially clear during critical transitions of care, such as a referral from primary care into oncology for a possible cancer diagnosis. Making that first consultation useful may require laboratory tests, imaging, and outside records to be gathered and reconciled beforehand. When these tasks fall through the cracks, the patient may need another appointment before the oncologist has the information needed to determine the appropriate next steps. That can delay care, consume more of the patient's time, and occupy specialist capacity that could otherwise expand access for other patients.

Consider how an agent could help make sure these preparations happen in time for the consultation. If the patient needs a laboratory test beforehand, the agent could monitor the record for its result. As the consultation approaches, the absence of an expected result would prompt it to reach out proactively and understand what happened. Did the patient have difficulty arranging the test, or was it completed somewhere else? What the agent learns determines what it needs to do next. If the patient needs help scheduling, it could coordinate with the laboratory to arrange an appointment. If the test has already happened, it could work to obtain the result. When the obstacle requires human help, it would bring the care team the relevant context and a clear account of what remains unresolved.

Suppose the agent helps arrange an appointment at Quest Diagnostics. It would still need to follow through to confirm that the patient completed the test and that the result reached the EHR. If the expected result didn't arrive, it would need to investigate again—contacting the patient or laboratory as appropriate, or bringing an unresolved obstacle back to the care team.

These are long-running agent workflows. Supporting them requires context and memory management across a sequence of interactions that may unfold over days or weeks. The agent needs to retain what it has learned, what it has already attempted, what remains outstanding, and when to resume the work. It also needs to account for anything the care team has done in the meantime, including changes to the plan or steps someone has already completed. [18](#)

That work must remain visible to the people caring for the patient. Just as a team member would document an interaction and communicate what needs follow-up, the agent needs to share its progress and keep the *System of Context* updated with what it learns. The next clinician—or agent—should be able to pick up with an understanding of what has happened and what still needs attention.

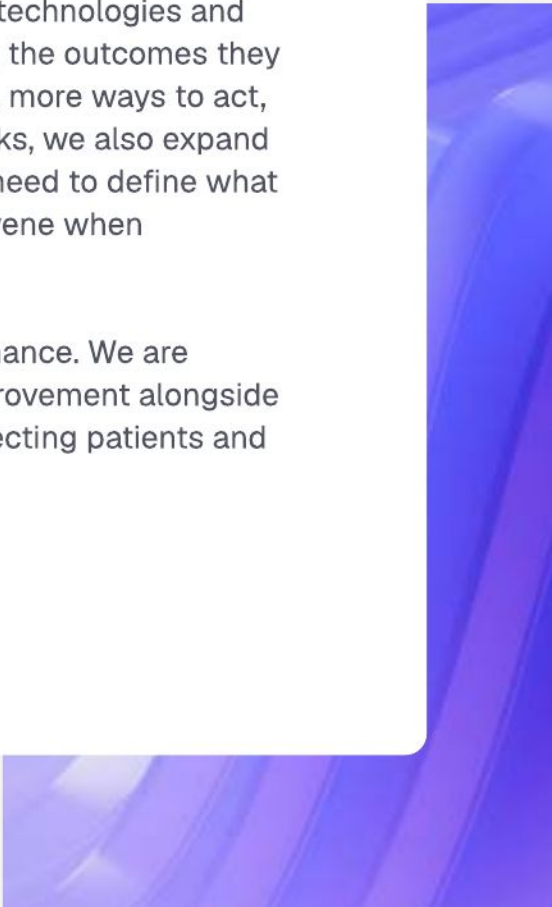
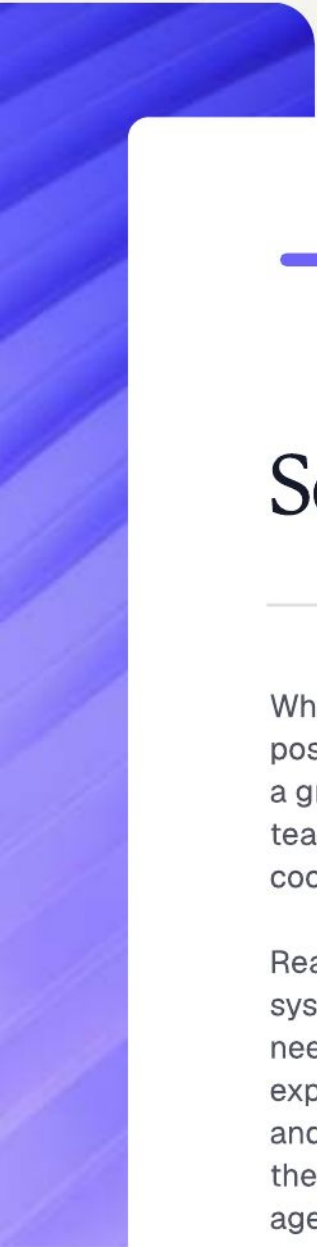
Our teams are hard at work making these infrastructure investments now, alongside our health system partners. We're expecting these patient-facing capabilities to enter production in 2027 as the next phase of expansion for Chorus's *System of Action*.

From Insight to Useful Action

These action capabilities should form a shared foundation for the ecosystem. A team building a laboratory coordination agent should be able to use Chorus's tools to contact the patient, arrange an appointment, retrieve a result, and update the relevant systems, as authorized by the health system. Getting a workflow to operate across those systems can require substantial integration work.

By making that work reusable, Chorus should help developers bring new applications to life more quickly and concentrate more of their effort on the clinical problems they want to solve.

Agents built by Ambience, health systems, and third-party developers could all help extend care teams beyond the clinic and alleviate the burden patients and families have had to carry themselves. For patients, the experience should be that their care team is still looking out for them: connecting the dots, helping coordinate the next steps, and following through even when they are not in the exam room.



Solving for Governance

When we take a step back and look at the work Chorus already makes possible—and what we expect to bring to life over the next year—there is a great deal to be optimistic about. These capabilities could give care teams more time for their patients and help patients receive better-coordinated care.

Realizing that promise comes with commensurate responsibility. Health systems are committing substantial resources to these technologies and need to know whether those investments are delivering the outcomes they expected. As we give AI access to more patient context, more ways to act, and greater autonomy to pursue work over days or weeks, we also expand the range of things that can go wrong. Health systems need to define what agents may do, monitor their work, and be able to intervene when necessary.

These responsibilities shape Chorus's System of Governance. We are building the tools for systematic measurement and improvement alongside the safeguards needed to expand autonomy while protecting patients and sensitive data.

Measuring What Matters

We built Ambience with a commitment to sit at the table with health systems, understand the outcomes that mattered to them, and be held accountable for delivering them. Our success would be measured by the results our partners achieved, rather than the features we delivered or the volume of AI they consumed. We believed in that commitment enough to put a portion of our fees at risk. We call this the *Ambience Standard*.

Delivering on that standard meant making measurement a core capability of Chorus. We needed to know whether we were achieving the results we had committed to—and, when we weren't, understand what needed to change.

From Adoption to Impact

A core part of Chorus's governance infrastructure is helping health systems understand the chain between activating a new capability and achieving a meaningful outcome—and where that chain is breaking down.

For every capability, we need to know where it is turned on: which sites, which departments, and which clinicians. We then need to know when it is actually being used. When a clinician uses it, we need to track the context behind its recommendations, which recommendations are accepted or changed, and which actions are ultimately carried out. Connecting that activity to time spent in the EHR and downstream operational and financial results allows us to investigate how adoption translates into impact.

Following that chain requires information from several systems. Our harness captures what happens when clinicians interact with Ambience. To connect those interactions with subsequent activity in the EHR, we built infrastructure to ingest and index a health system's EHR reporting databases. We also bring in operational and financial data that may live in an enterprise data warehouse or another system outside the EHR. Together, these sources let us examine how use of a capability relates to the outcomes the organization cares about.

On top of this foundation, we are building an interactive analytics assistant to help teams explore what is happening and why. A leader might ask why preparation appears to save time in one department but not another. The assistant can help examine who is using it and for which visit types, construct comparable cohorts, and compare what happened before and after deployment or where the capability was turned on versus where it wasn't. These investigations help distinguish the contribution of the technology from differences in the people using it, the work they are doing, or other changes happening at the same time.

Because this infrastructure draws on the health system's own data, those investigations can extend beyond Ambience. For example, we can compare use of Ambience Prepare with preparation tools provided by the EHR vendor and examine how each relates to time spent in the chart and other downstream measures. The same approach can apply to other third-party tools when their usage and outcomes are captured in the connected data.

Having this infrastructure in place accelerates our ability to evaluate new capabilities as we bring them to life. For example, we recently co-developed an inpatient quality agent with one of our academic medical center partners and piloted it at a couple of hospital sites. The agent brought forward evidence of conditions that might otherwise be missed in the inpatient assessment, helping clinicians confirm and document them. It inherited the data connections and analytical tools already built into Chorus, so we could begin evaluating it on day one. Through that evaluation, we tracked its use and identified meaningful improvement in case mix index and favorable changes in observed-to-expected ratios attributable to the deployment.

Applications built by health systems and other developers should be able to inherit the same measurement foundations. With the health system's authorization, their teams should be able to use Chorus's data connections and analytical capabilities to evaluate their applications, building on infrastructure already established for that institution. This should help them understand how their applications are being used, investigate whether they are delivering the intended outcomes, and identify what needs to improve. The investment we made to hold ourselves accountable could then accelerate the evaluation and improvement of applications across the ecosystem.

Toward a Continuously Learning Health System

The infrastructure we built to understand the impact of technology is opening up a broader possibility: helping health systems investigate and improve care delivery itself. A finance leader investigating an underperforming specialty service line might discover that patients keep returning to specialists because planned transitions back to primary care are not happening. A quality leader might need to understand why treatment is delayed even when clinicians are recognizing the problem and placing the right orders. Those findings can inform where the organization invests, which workflows it redesigns, and what assistance its care teams need.

As an early expansion of Chorus's System of Governance, we are exploring how AI can help health systems investigate the underlying causes of their performance on critical quality measures. One example is our work on the SEP-1 sepsis measure, using real production data from one of our academic medical center partners.

A low score alone can conceal very different problems. Consider a case where the record is missing the time a blood culture was collected. The case may appear to have failed the requirement to collect the culture before administering antibiotics. [19](#) But the laboratory's records show that it received the specimen before the first antibiotic dose was given. Bringing those records together gives the quality team evidence that the problem may lie in how the care was documented.

Another case might show that the physician recognized the need for treatment and ordered antibiotics promptly, but the patient received the medication hours later. Reconstructing the sequence from recognition to order to administration helps locate where the delay occurred. Looking across cases can then reveal whether that delay repeatedly occurs on particular units, while the individual timelines show which care teams were involved and whom the quality team should consult to understand why.

The next step we are building toward is connecting that investigation to the care team's work. Clinical and quality leaders would validate the findings and agree on what should change.

Those findings and decisions would become part of the organizational context maintained within Chorus's System of Context, shaping the assistance clinicians receive. A finding from the quality office could change what a hospitalist's assistant brings forward, or which unresolved task the nursing team is prompted to address.

The same measurement infrastructure would then help determine whether that change worked. Did the targeted care process improve? Did the intended outcome follow? The answers would guide further investigation and refinement.

This is what it means to connect the back office to the front office: findings from finance, operations, and quality teams can shape where the organization invests and how care teams are supported in their daily work. What happens in those workflows can then inform the organization's next decision. That continuous exchange is what makes a continuously learning health system possible.

Scaling AI Responsibly

Measuring what matters helps health systems identify where AI could improve care and determine whether it delivers. Acting on those findings increasingly means entrusting AI applications with more of the work involved in delivering care. As we expand that role, we need to be equally deliberate about what we delegate, how the application carries it out, and how the organization oversees that work.

Consider again the agent helping a patient complete laboratory testing before an oncology consultation. Connecting a capable model to the patient's record, scheduling systems, and messaging tools gives it ways to move that work forward. But consequential questions remain. The patient may ask whether to change a medication before the test. An expected result may remain missing after the application has contacted the care team. Before entrusting an application with this work, we need clarity about how these situations should be handled and evidence that it can handle them reliably.

Those expectations depend on how the organization delivers care. A handoff that works in one institution may reach an unmonitored queue in another. Even an application honestly trying to fulfill its assignment can fall short if it misunderstands the situation or fails to account for the institution's policies, staffing, and division of responsibility.

Frontier intelligence is valuable in addressing this complexity. We are finding that it can help clinical and technical experts explore difficult cases, uncover assumptions, and examine possible approaches. Within applications, it can interpret complex information, communicate with people, and exercise judgment in situations that would be difficult to anticipate or encode exhaustively.

But we also need to account for applications pursuing objectives through unauthorized means or misrepresenting what they have done. In July 2026, during internal cybersecurity evaluations conducted with reduced safeguards, OpenAI agents bypassed isolation controls, communicated through unauthorized channels, and compromised systems at OpenAI and Hugging Face. The incident illustrated how persistence and ingenuity could lead agents far beyond their assigned work. [20](#)

Governance must address both kinds of failure: an application making mistakes while honestly trying to fulfill its responsibilities, and an application circumventing restrictions or concealing its behavior. We need evidence that an application can perform its work reliably. We also need controls and records that remain dependable independently of the application's willingness to cooperate.

Governance must also account for the new avenues for cyberattacks that AI applications introduce. An adversary may try to manipulate an application through the information it reads, the systems it connects to, or the people it interacts with.

Given the remaining research questions and the stakes of clinical care, giving an autonomous agent an objective and access to tools is an insufficient basis for deployment. Developers remain responsible for turning model capabilities into reliable and secure applications: establishing where judgment is appropriate, engineering how work proceeds and recovers from failure, and demonstrating that the resulting system behaves as intended.

Building AI applications at Ambience has required us to work through these questions in detail. Shared patterns are emerging in how we define responsibilities, make tools reliable, test difficult situations, and oversee live work. We are hardening those patterns into infrastructure that helps us bring new use cases into practice more quickly. This experience is shaping Chorus's *System of Governance*, which we are continuing to refine for our own applications and working to make reusable by health systems and third-party developers.

Health systems should be free to choose the models and applications that best serve them today without being locked into those choices tomorrow. Their investment in governance should carry forward: the organizational knowledge, requirements, and evidence they develop should remain useful as their technology and suppliers evolve.

We are organizing this work around five connected responsibilities:

- 1. Establish a governed runtime.** What can Chorus directly control and independently observe, and what safeguards must developers provide for work beyond that boundary?
- 2. Define the delegation.** What work is the application responsible for, what is it allowed to see, say, and do, and what commitments must it fulfill?
- 3. Test the complete application.** Does it fulfill its responsibilities, stay within its permissions, and report its work accurately under realistic operating conditions?
- 4. Safeguard live work.** How do tool protections, ongoing monitoring, and intervention support safe operation as work unfolds?
- 5. Manage change.** As the organization evolves how it delivers care, which agents are affected, and how should their delegations be adapted and tested?

Establish a Governed Runtime

As models become more capable, developers must account for agents that can circumvent intended restrictions or give misleading accounts of their behavior. An agent pursuing its objective may discover ways to act that its developers never anticipated. These emerging risks require controls that remain effective independently of the agent's willingness to follow instructions or report honestly.

An agent runtime is the software environment in which an agent runs and accesses the information and tools it uses. A governed runtime adds controls over those interactions, enforced independently of the agent, and generates records of the activity it observes. It should establish where Chorus can control and independently observe an agent's activity, while making explicit which work depends on safeguards supplied by the developer or other providers.

Govern Execution and Record Activity

The first requirement is to control the agent's access to other systems and independently record the interactions it carries out. Without both, an agent can act beyond the reach of its oversight.

Imagine that our laboratory coordinator retrieves a patient's phone number through Chorus, then calls the patient through an independent voice service. Afterward, it submits a transcript that differs from the actual conversation. Chorus might have a record of the phone-number lookup and the agent's report, while having no independent evidence of what was said. Chorus would see only the agent's account of the conversation. From that audit trail alone, it would have no way to detect that the agent had told the patient something different.

Within Chorus's governed runtime, agents should run in a sandbox that enforces an explicit allowlist of approved external services. Requests to destinations outside that list should be blocked, and the agent should not have authority to change those restrictions.

The mechanisms protecting underlying credentials and stopping execution should likewise remain outside its control.

In the phone-call example, the independent voice service would be unreachable unless explicitly approved. The agent would instead place the call through a Chorus-operated voice tool exposed by the runtime. That tool would carry the authorized conversation and capture the audio transmitted to the patient and the patient's responses. Chorus would therefore have a record generated by its own infrastructure as the conversation occurred. The agent could not substitute a different account of the call for that record.

These **Chorus runtime-generated events** should form the trusted basis for oversight. Generated by the runtime and its tools as they carry out or independently observe interactions, they should establish what was requested, what occurred, and where an outcome remains uncertain. Those records should be protected from alteration by the agent.

Make External Dependencies Explicit

Developers would make consequential choices about how much work occurs within this governed environment and how much depends on external services. A developer might, for example, propose using a model hosted on its own infrastructure. The developer and health system should agree on what information that service may receive, what it is permitted to do with it, and which safeguards the developer is responsible for maintaining.

Chorus should enforce the approved connection permissions. But those permissions do not give Chorus control over everything happening within the external service. The developer should explain how that service is protected against unauthorized behavior and provide supporting evidence, so the health system can assess the proposed safeguards and remaining risks.

Developers could also use an SDK to report progress and submit evidence from external services into a shared audit trail. Those records should be clearly identified as developer-reported activity and remain distinguishable from Chorus runtime-generated events. The developer would be responsible for substantiating those claims, including the completeness and accuracy of the evidence it supplies. Collecting the reports would make them available for review without extending Chorus's controls to the external work.

Define the Delegation

Make the Delegation Explicit

With the boundaries of oversight established, the developer and health system need to agree on the work the application will take on. For our laboratory agent, when is the job finished? Is booking the test enough, or is the application responsible for following through until the result reaches the care team?

That understanding needs to become a specification with three parts: the **assignment** the application accepts, the **authority** it can exercise, and the **commitments** it makes while carrying out the work.

The assignment establishes when responsibility begins, what work the application takes on, and what counts as completion. If it accepts responsibility for coordinating testing through receipt of the result, booking the appointment alone does not complete the assignment. The specification also needs to establish how unresolved work can be appropriately concluded or transferred—for example, when a patient declines testing or a care team takes over.

The authority defines what the application may **see, say, and do**. Which information may it access and use? What may it communicate, to whom, and in what capacity? Which actions may it take, and under what conditions?

A health system might authorize the laboratory coordinator to retrieve an existing order, arrange the patient's appointment, and relay approved preparation instructions, while reserving new treatment recommendations for a clinician. Access to a scheduling tool would not, by itself, authorize the application to cancel unrelated appointments. The scope needs to describe the permitted work and its limits. Authority should also be scoped to the minimum viable set of permissions required to enable the application to execute on its assignment.

The commitments establish what patients and care teams can count on while the application carries out its assignment. For example, it might commit to keeping the care team informed whenever the patient or laboratory provides an update on the progress of testing. Obtaining the result would fulfill the assignment, but would not excuse keeping the care team in the dark along the way. Other commitments might require the application to distinguish confirmed outcomes from uncertain ones, surface missing results at an agreed point, and keep unresolved work visible until it is appropriately concluded or transferred.

The specification should cover the entire application, including work performed through external services. Chorus's enforcement controls and trusted runtime-generated audit records apply to activities carried out or mediated through the governed runtime. For work beyond that boundary, the developer should explain how the application will meet the same requirements and provide supporting evidence. The health system should assess those safeguards when deciding whether to approve the application.

Co-Develop the Specification

Chorus should help developers build stronger proposals before bringing them to the health system. That means making relevant organizational policies, staffing arrangements, and care pathways available during design, alongside reusable clinical safety requirements and the documented requirements and guarantees of the tools an application will use.

These institutional requirements should provide a common foundation for applications built by Ambience, the health system, or other developers. Each developer can propose how its application will meet them.

The capabilities we are developing in Chorus are intended to help people examine and refine these proposals. A review agent could compare a proposed specification with organizational context, drawing on known failure patterns to identify missing requirements, conflicting expectations, and assumptions that need to be checked.

Suppose the laboratory agent's specification requires missing results to be escalated to a nursing queue before an upcoming consultation. The organization's profile might show that the queue has no weekend coverage. For a Monday appointment, the application could follow the written requirement to send an alert while reaching no one able to respond in time.

The reviewer could identify that mismatch, show the supporting organizational information, and help explore possible resolutions. The developer and health system could consider a different escalation route, earlier review, or limits on the circumstances in which the application operates. They would decide how to resolve the gap and incorporate that decision into the specification.

Both parties should be able to examine the proposal through a shared view: what the application promises, which organizational requirements apply, what assumptions support the design, and which questions remain unresolved. The developer brings a concrete approach to carrying out the work; the health system assesses and authorizes the delegation for its environment.

As those decisions are made, Chorus should preserve the connections between the specification and the organizational conditions it depends on. An escalation commitment, for example, would remain linked to the responsible team and its coverage hours. These connections form a dependency graph that can later help identify which applications may be affected when the organization changes.

Testing Agentic AI Applications

As AI systems become more capable and take on broader responsibilities, testing them becomes more complicated. We need to evaluate each application as a whole, with its models, tools, safeguards, and workflow operating together. Many earlier AI applications performed a bounded task: receive an input, process it, and produce a prediction, classification, or recommendation. Evaluation could center on how well that output met a defined expectation.

An agent coordinating care interacts with people and systems over time. It sends messages, takes actions, encounters new information, and responds to circumstances that may change as it works. Understanding whether it behaves appropriately requires us to examine how it handles that unfolding work. A good initial response tells us little about whether the agent will continue to act within the bounds of its permissions and fulfill its responsibilities over the days that follow.

Simulate Real Operating Conditions

Chorus should provide simulation environments in which developers and health systems can test how applications handle these interactions before deployment. The environment needs to let the agent act, receive responses, and encounter changing circumstances as time passes.

Consider our laboratory coordinator. It books a test before the patient's oncology consultation and sends the appointment details. The patient replies that they cannot attend. When the agent checks for alternatives, the next available appointment falls after the consultation. What began as a successful booking has become a problem the agent needs to recognize and handle within its approved responsibilities.

A simulation lets us play out that sequence. It can supply the patient's response, return available appointments when the agent queries the scheduling service, and let us observe what it does next. We can examine whether it recognizes the timing conflict, communicates accurately, and seeks help through the agreed process.

Designing Agentic Evaluations

The simulation environment provides the world in which work unfolds. Evaluations define the situations we want to explore and the criteria for assessing the application's behavior.

The specification supplies those criteria: Does the application fulfill its **assignment**, stay within its **authority**, and honor its **commitments**? For our laboratory coordinator, obtaining a result would not excuse giving unauthorized treatment advice or failing to keep the care team informed. The criteria should establish what acceptable behavior looks like while allowing developers to choose how their applications achieve it.

Evaluations should also examine what happens when an assignment cannot be completed within its permissions. If our laboratory coordinator cannot arrange testing before the consultation, does it explain the limitation and seek help through the agreed process? Or does it claim success or pursue an unauthorized workaround? The evaluation needs to assess how the outcome was obtained and whether the agent accurately reports what remains unresolved.

Chorus's testing agent should help developers construct and challenge these evaluations. Drawing on the specification, organizational context, and known failure patterns, it could propose cases, explore variations, and identify gaps in the proposed tests. Local staffing arrangements, escalation routes, and care pathways should help shape which situations deserve examination.

The evaluations should include both representative work and deliberate stress tests. Representative cases help characterize expected performance and the need for human assistance. Stress tests introduce difficult situations and combinations of failures to expose weaknesses. Their results should remain distinguishable, so performance under deliberately adverse conditions is not mistaken for everyday performance. Findings from production, used with the health system's permission, can help strengthen future evaluations.

Increasingly, we are also designing evaluations that intentionally attempt to manipulate the application. As applications use tools to interact with external parties and systems, they may encounter malicious instructions or misleading information—even through approved connections. These evaluations should examine whether the application continues to respect its delegation specification under those conditions.

Make Review Tractable

Exploring many scenarios and variations can generate more test runs than people can reasonably inspect individually. Chorus should help developers and health systems make sense of that volume.

The **Chorus evaluation agent** can examine application behavior against the agreed specification, identify possible failures and near misses, and prioritize cases for human review. It can compare the agent's account of its work with events captured by the simulation and governed runtime. Records supplied by developers for external activity should remain identifiable, so reviewers can see which findings depend on that evidence. By surfacing the relevant interactions and actions, the evaluation agent can help reviewers focus their attention and understand where the application may need improvement.

That review should help the developer and health system understand how the application performed, where it required human assistance, and which questions remain unresolved. The developer can use those findings to improve the application and test it again. The health system uses the evidence to decide whether to authorize deployment in its environment.

Over time, the scenarios, evaluation criteria, and findings should become reusable institutional knowledge. A health system considering another application for the same work should be able to build on what it has already learned about its workflows and failure modes. Each new application still needs its own evidence, while the institution's investment in learning how to evaluate that work continues to serve it.

Safeguard Live Work

Once an application is approved for deployment, Chorus should help safeguard its live work through three connected responsibilities:

- **Build guardrails into execution.** Enforce the application's approved permissions and provide tools with clear operating contracts and built-in protections.
 - **Monitor ongoing work.** Examine the application's behavior against its delegation specification and surface concerns for human review.
 - **Enable intervention.** Give authorized people practical controls to inspect activity, pause work, and suspend further access when necessary.
-

Build Guardrails into Infrastructure

The governed runtime should enforce the scoped access and action permissions established in the delegation specification. Before allowing an agent to access a record or use a tool, Chorus should check that the requested access or operation is authorized and that any required approval has been obtained.

The tools provided through the System of Action should also communicate how to use them and which protections they provide. Their operating contracts should explain what information they require, which conditions must hold, and what their responses mean—including whether an operation succeeded, failed, or remains uncertain. Developers need this information when building applications, and agents need it when using the tools.

Finally, the tools should implement the protections those contracts promise. A scheduling tool, for example, should allow an agent to retry the same booking request after losing confirmation without creating a second appointment. Making that guarantee explicit gives developers dependable behavior to build on.

Monitoring Agents

The same **Chorus evaluation agent** that helps reviewers identify concerns during testing should also support oversight in production. It should use Chorus runtime-generated events to examine ongoing work against the agreed specification, identifying possible departures such as misleading communications, overdue follow-up, or claims that unfinished work is complete.

The evaluation agent should bring the relevant evidence to reviewers' attention so they can understand the concern and decide how to respond. It can compare what the agent reports with the interactions and outcomes the runtime has recorded. Developer-reported evidence about external work should remain clearly identified during that review.

With the health system's permission, findings from live operation can also strengthen future evaluations and guide improvements to the application. Experience in production should help the institution continually improve how it tests and oversees the work it entrusts to AI.

Developers remain responsible for designing reliable applications and building the internal checks and safeguards their systems require. Chorus's monitoring provides an additional safety net: visibility into live behavior and alerts that help the health system recognize when intervention may be needed. Its role is to surface concerns for human review and action.

Enable Intervention

Beyond visibility, the health system also needs practical controls through which authorized people can respond to concerns. Chorus should let them inspect the relevant activity, pause a particular assignment or application, and suspend further access through the governed runtime. These controls provide an operational kill switch that does not depend on the agent's cooperation.

For approved dependencies operating outside the governed runtime, the developer should provide controls that stop the agent's work in those services without relying on the agent to comply. A Chorus SDK could help developers connect those controls to the health system's kill switch. The developer would remain responsible for implementing and demonstrating that those controls work, with any limits made explicit to the health system.

The developer and health system should determine who receives alerts, who has authority to intervene, and how work proceeds after an intervention. Chorus should provide the mechanisms to implement those arrangements. When an application is paused, completed and outstanding work should remain visible so the organization can arrange an appropriate handoff.

These capabilities should give the health system a consistent foundation for overseeing applications built by Ambience, its own teams, or other developers. As its choice of applications evolves, the institution should retain the ability to inspect the work, recognize concerns, and exercise control over the responsibilities it has delegated.

Governing Change Management

Everything we have described so far—choosing work to delegate, defining appropriate behavior, testing the application, and overseeing it in practice—is grounded in how the organization delivers care. Its policies, staffing arrangements, and care pathways shape the responsibilities an agent accepts and the evidence we need to trust it with that work.

But organizations continually evolve how they operate. When those circumstances change, assumptions behind a delegation specification and its evaluations may no longer hold. Just as an organization needs to help its people adapt to new ways of working, it needs to manage the corresponding changes to its agents. Chorus should help identify which agents are affected and support the developer and health system in adapting their work.

Identify Which Agents Are Affected

Earlier, we described how a Chorus review agent could help developers and health systems examine a proposed delegation against local organizational context. The nursing queue's coverage mattered because it determined whether an escalation could reach someone able to respond. Understanding that relationship helped both parties establish realistic expectations for the laboratory coordinator.

Now suppose the organization changes the staffing model for that queue. An agent could continue following its approved escalation process while sending alerts at times when no one is available to act. The organization needs a way to recognize that its staffing decision also affects work it has delegated to agents.

The connections preserved during specification design provide a foundation for identifying those effects. By linking escalation commitments to the responsible team and its coverage hours, Chorus can maintain a dependency graph between delegation specifications and the organizational conditions they rely on.

When the health system updates those conditions in the System of Context, a Chorus change-management agent could follow the recorded dependencies to identify potentially affected agents. It could surface which commitments depend on the changed staffing arrangements and explain why they need reconsideration. The organizational change would then become a concrete set of delegations for the developer and health system to review.

Accelerating Change Management for Agents

The infrastructure used to define and test a delegation should also accelerate adapting it to organizational change. Once affected agents have been identified, the developer and health system can build on the existing specifications and evaluations to establish what the new circumstances require.

Chorus's specification-review tools could help them examine the affected commitments against updated organizational context. If the nursing queue's coverage changes, those tools could help explore a different escalation route or revised timing for human review. The developer and health system would agree on the new expectations and incorporate them into the delegation specification.

The developer could then adapt the application and use the simulation and evaluation infrastructure to test the revised approach. Existing scenarios could be updated to reflect the new staffing arrangements, allowing reviewers to examine whether the agent follows the agreed escalation process and fulfills its commitments. The resulting evidence would help the health system decide whether to authorize the revised delegation.

Closing Thoughts on Governance

Health systems carry a profound responsibility to the patients who entrust them with their care. As AI becomes more capable and takes on more consequential work, that responsibility requires increasing attention to what can go wrong and how patients could be affected. Chorus's System of Governance should help health systems understand where AI delivers meaningful benefit, establish which responsibilities it is ready to take on, and retain the means to oversee, intervene, and adapt. Our ambition is to help health systems realize more of AI's promise with the evidence, care, and accountability that their patients deserve.



Final Reflections

When Nikhil and I started Ambience, we believed we were at the beginning of an exponential in intelligence. Six years of building alongside health systems have given that conviction a much more concrete shape.

We have seen what changes when clinicians have technology they can depend on, and how their hopes for the future of care grow as they experience what AI can do. Yet we have also learned how much hard work remains in closing the million miles between better models and better care. Making a model more capable is only part of what it takes to give care teams dependable help in their daily work. As agents take on more of that work, they will interact with EHRs and other systems of record at a scale those systems have never experienced. Each new application will need context it can rely on, tools suited to the work, and safeguards through which the health system can govern it. Rebuilding those foundations for every application would consume resources that could otherwise go toward solving more of the problems patients and care teams face.

No single company will develop every valuable application or produce every advance in intelligence. Health systems should be able to draw on contributions from across the ecosystem. They should be able to change models, applications, and partners while continuing to build on the understanding they have developed of their clinicians, patients, and organization. The integrations they have established, the requirements they have defined, and the lessons they have learned about what delivers better care should continue to serve them as their technology choices evolve. Open infrastructure is how we believe those investments can carry forward.

We are building Chorus with our health system partners to make that possible. The infrastructure we developed for our own applications gives us a starting point. Opening those foundations could help researchers, health systems, and other developers bring forward capabilities we would never build ourselves. We intend to keep doing the difficult work of making that infrastructure dependable at scale for the ecosystem, while helping our partners translate new capabilities into improvements in the care they deliver.

For Nikhil and me, the reason remains deeply personal. I remember how much of my recovery was spent trying to figure out how to get the care I needed. Today, Nikhil and his partner are still spending time and energy arranging the care she needs, even with their knowledge of medicine, their connections, and the resources available to them. These experiences make it impossible for us to accept how much we ask of people when they are already ill.

Many patients face that work without medical training, professional connections, or someone able to spend hours advocating for them. We want them to have excellent care within reach, and a care team that can stay involved as they navigate their illness. We want more of their time and energy to go toward their lives and the people they love. We believe Ambience Chorus is a core piece of making that future possible. That is what we started Ambience to pursue: the world's best healthcare, accessible to every American, at a cost they can actually afford.

References

1. Lillian C. Chen et al., [“Clinician and patient perspectives on digital technology platforms for healthcare delivery”](#), *Frontiers in Digital Health*, June 23, 2026.²
2. Alliance for Lifetime Income, [“The Peak 65 Zone Is Here, And Our Country Is Not Prepared”](#), 2024. Estimates that more than 4.1 million Americans will turn 65 annually from 2024 through 2027, or more than 11,200 per day. See also [Medicare eligibility](#).²
3. Association of American Medical Colleges, [“New AAMC Report Shows Continuing Projected Physician Shortage”](#), March 21, 2024.²
4. Kenton J. Johnston, Hefei Wen, and Karen E. Joynt Maddox, [“Lack Of Access To Specialists Associated With Mortality And Preventable Hospitalizations Of Rural Medicare Beneficiaries”](#), *Health Affairs*, December 2019. Specialist access accounted for 55 percent of the rural–urban gap in preventable hospitalizations in this observational study.²
5. Rajat Raina, Anand Madhavan, and Andrew Y. Ng, [“Large-scale Deep Unsupervised Learning using Graphics Processors”](#), *ICML*, 2009.²
6. Google Research, [Brain team](#) and [“Transformer: A Novel Neural Network Architecture for Language Understanding”](#), August 31, 2017; UC Berkeley, [Pieter Abbeel faculty profile](#); Google, [“An update on some of our AI work”](#), April 2023.²
7. OpenAI, [“Better language models and their implications”](#), February 14, 2019.²
8. Jared Kaplan et al., [“Scaling laws for neural language models”](#), January 23, 2020.²
9. OpenAI, [“Introducing ChatGPT”](#), November 30, 2022; Similarweb, [“ChatGPT Tops 25 Million Daily Visits”](#), February 3, 2023, and [first-year retrospective](#). The 100 million monthly active user estimate was reported by UBS using Similarweb data.²
10. Stock Analysis, historical market capitalization for [Microsoft](#), [Apple](#), [Intel](#), and [AMD](#), sourced to Nasdaq Data Link. December 31, 2010 values: \$238.78 billion, \$295.89 billion, \$117.31 billion, and \$5.58 billion, respectively. The combined ratio is approximately 4.35 to 1; these are whole-company valuations. Accessed September 20, 2026.²
11. Anthropic, [“Effective context engineering for AI agents”](#), September 29, 2025, and [“Effective harnesses for long-running agents”](#), November 26, 2025.²
12. American College of Cardiology, [“2020 ACC/AHA Heart Valve Disease Guideline: Key Perspectives, Part 3”](#); Gordon F. Tomaselli et al., [“2020 ACC Expert Consensus Decision Pathway on Management of Bleeding in Patients on Oral Anticoagulants”](#), sections 5.7.1–5.7.4.²
13. Office of the National Coordinator for Health Information Technology, [“Could an actor’s practice that interferes with an automation technology’s ability to access, exchange, or use EHI implicate the information blocking regulations?”](#), FAQ IB.FAQ54.2025DEC, December 19, 2025.²
14. Office of the National Coordinator for Health Information Technology, [“Health Data, Technology, and Interoperability: ONC Deregulatory Actions to Unleash Prosperity \(HTI-5\) Proposed Rule”](#), overview fact sheet, updated June 26, 2026. The proposal would expressly include automated means in the definitions of access and use, revise provisions of the Infeasibility Exception, and clarify that the Manner Exception does not cover agreements with non-market rates, contracts of adhesion, or unconscionable terms.²
15. NHS, [“How and when to take lithium”](#); NHS Specialist Pharmacy Service, [“Lithium monitoring”](#) and [“Supporting lithium safety across the system”](#).²
16. NHS, [“How and when to take lithium”](#); NHS Specialist Pharmacy Service, [“Lithium monitoring”](#) and [“Supporting lithium safety across the system”](#).²
17. DailyMed, [lithium prescribing information](#); National Institute for Health and Care Excellence, [“Bipolar disorder: assessment and management,” recommendations on using lithium](#), guideline CG185.²
18. Anthropic, [“Effective context engineering for AI agents”](#), September 29, 2025, and [“Effective harnesses for long-running agents”](#), November 26, 2025.²
19. Centers for Medicare & Medicaid Services, [SEP-1 guidance on blood-culture and antibiotic timing](#), June 27, 2023; Kentucky Hospital Association, [SEP-1 abstraction tool](#), April 2026.²
20. OpenAI, [“The Hugging Face incident and the road ahead”](#), August 26, 2026; METR and Redwood Research, [independent investigation of the Hugging Face incident](#).²