Elion

Buyer Review Transcript

Product Reviewed	Ribbon
Integrations Reviewed	N/A
Other Products Mentioned	N/A
Date	07/06/2023
Expert Role	Software Engineer
Care Model	Consumer-Focused Health Insurance Plan
Payment Model	Fee for service

Today we're going to be talking about Ribbon and how it's being used at your company. Before we jump into that, could you give a brief overview of your company and your role there?

We're building a consumer-focused health insurance plan. I'm a software engineer at the company.

When did you purchase Ribbon and how long have you been using it?

We purchased Ribbon around July or August 2022 and have been using it ever since. So about a year now.

Can you tell me about the Ribbon product and how you use it?

The Ribbon product helps clean a lot of our provider and location data for our network directory, and then surfaces that information through an API after their process runs.

What that looks like is they help us provide metadata about a provider using an identifier like the NPI (National Provider Identifier). We don't have to verify which information source is the accurate one – they do that for us. We then consume their API in order to surface that data into our network directory.

For your network directory, is the use of that data primarily to enable your members to find care? Or are you using it for internal purposes as well? In other words, what are the primary use cases of the data that you're pulling from them?

Yeah, the primary use case is what you said, allowing our members and potential members to examine how wide our network is and if it's adequate for their needs.

Since we started using Ribbon, there's also other use cases. It makes it easier to export data to a vendor that needs to be in a certain format. It also makes it easier for us to identify steerage opportunities based on some of the filters that Ribbon can support on top of data that they surface.

With regard to exporting data to another vendor – do you have an example of what that looks like in practice?

So for example, one of our vendors helps us to process our claims. One thing they need to know is whether they should include or exclude a certain provider for certain types of claims, depending on what plan our member has purchased. In that specific case, we need to provide them a list of what providers we excluded and why so that they can better make that decision themselves when they're processing a new claim. Additionally, we can help our customer service agents better guide our members by allowing an export of filtered search results (built on top of the Ribbon API), which can also be sent along to any other providers, vendors, or partners that might need it.

One thing Ribbon does, along with providing additional data, is they bucket our providers into specialties. And so we can carve out certain parts of the network for certain insurance plans that we offer by using their specialty classification product. We can hit their API and pull providers with those specialties and then easily combine that into lists that we can give to our TPA (Third Party Administrator).

Is the right way to think about evaluating Ribbon in terms of the following: 1) the quality of the data that you are getting from them, and 2) how easy it is to work with their API? I'm not hearing that you're using any sort of UI or application.

On the UI piece, Ribbon does have a white labeled UI as something that someone can purchase, but we decided not to do that. This was mainly because of our model, where we want to surface particular providers in a different way than their other clients.

And their UI, is that specifically a provider network application?

That's right – provider search.

Let’s dig into each of the other two elements individually – their data quality and how easy it is to build against their API. First let's start with the data. What data specifically are you getting? And what have you found in terms of the quality and usefulness of that data?

They do a lot of scraping, for example, to tell us a provider's education that we can surface in a UI. Also, they attach specialties to a provider that we can surface. Those are the main things that we use that would be difficult for us to do. So they help answer certain demographic and “what kind of care they provide” questions, so that we don’t have to dig for this information.

They also can surface insurance plans that a provider takes, or locations that they're affiliated with. But given the nature of our products and our network, we don't use those. Our main source of data comes directly from our wrap network and we only send them locations that in-network providers practice at based on that data. So all this is to say that Ribbon does not really add any value for us in terms of, “Oh, we didn't know that this provider practiced at this location.” We've chosen not to allow those to be in our Ribbon instance, because we would want to avoid any inaccuracies. Essentially, we're leaning on our wrap network as our source of truth for what is in-network or not.

Ribbon also quality checks our location data. They do a lot of geo matching to make sure, based on the location we sent, whether the address is the same or not. For example, if the wrap network data contains multiple addresses that, based on a string match, might be different, they help to resolve that.

Take the address 14 West Ninth Street, Suite 205. There might be an additional space or not, or street might be spelled “Street” or “St.” They have something in their process to reduce that location into one. They take care of situations where our wrap network data might associate five providers with the location that has “Street” spelled out, and another ten providers with the location where street is spelled “St.” That would be a big pain point if we had to resolve that. But they do that for us, which is a big value add.

They also provide latitude and longitude coordinates with any location so that we can easily show it on our map and our user interface. Additionally they provide a distance value, so we can sort providers and locations by distance from a specific zip code. So that's something nice that we can use to help our members find the closest provider or location that they can go to.

They also provide a confidence score. That is, based on their data, they can provide their confidence in the data being correct. We thought about using this but we decided not to since our data truth leans on our wrap network files. But that's something that other people may consider using when looking at Ribbon.

Let me play that back to make sure I caught it. It sounds like the main data that you are using from Ribbon is the education and specialty data. And then your wrap network is the source of truth for everything else. However the wrap network data is a little bit messy, and so you send it to Ribbon, which helps consolidate and clean the data for use in your provider search tool.

That's right. I'm going to pull up an example to make more concrete what we actually send to Ribbon.

We send them a list of provider NPIs, as well as a locations file with what we believe to be the address. We also tell them which providers practice at which locations.

Ribbon takes this data and does the following: They augment the provider data with education and specialty data. They take the locations data and resolve it, as we just discussed. And then they update the provider-to-location linkages. They also add additional location details, such as phone numbers.

How would you characterize the quality of the data you're getting from them?

I would say the quality meets expectations. It's hard for us to actually know the quality of the data they're giving us because we largely surface it in our product.

I do remember when we were integrating with them in the beginning, there were a couple instances where they had some incorrect education data. They also thought that one provider was actually some other provider and incorrectly populated their information with data from a provider that wasn't even in our state. So there was some back and forth in the beginning about accuracy of the data. It was a little bit of a concern for us in the sense of, "Oh, if these are wrong, how many more are wrong?" But since then, purely experientially, we haven't really had any inaccuracies emerge (this is probably over eight months at this point). We don't have any QA checks that we run on their data or anything though.

It sounds like the way you would know about issues is if you were hearing from your members that some of the data is inaccurate.

That's right.

Related to that though – and I don't know if this is a specific Ribbon issue, an issue with our wrap network data, or just an issue with healthcare data in the US – there have been some members that have said that many of the provider phone numbers don't work, or have really long wait times. This is something that we don't have a way to improve, and is probably a really difficult problem to solve. So that's something we've observed.

A couple of months ago, Ribbon came out with a product where you can make appointments with certain locations or providers inside of their database. We haven't explored it, however. To be frank, this is mainly because we don't really trust the quality of the contact information in the product. This is one of those situations where if the first two or three members that try to use the product end up not able to make an appointment, or make an appointment and show up and can't get seen, that's a terrible experience. So for that reason we haven't really looked into this feature.

To play it back again, it sounds like, from what you can tell, the data quality has been pretty good (other than some hiccups at the beginning). You don't have a way of truly

validating that. But anecdotally, based on what you've heard, you're not seeing a lot of data issues being surfaced by your members.

That's right.

Anything else to say about data quality?

The only other thing I'll say is that since we started working with them – I believe in late Q1 or early Q2 of this year – they did a pretty big overhaul of some of their internal systems. That might be something worth looking into if you're considering Ribbon.

To double click there, did you notice improvements based on the work that they did?

Yeah, there were a couple of things. One is that we had some additional data we wanted to get surfaced back to us in order to better educate members who were navigating the directory. For example, we wanted to flag whether a provider is part of our wrap network or whether they are directly contracted. This was not possible with the Ribbon system until late Q1 or early Q2. Now that data has surfaced to us and we can use it in our UI.

Related to that, I also asked them about a certain use case of a directly contracted provider that used to be associated with just one location, but now is associated with two. I wanted to make sure that the two different geographies would be resolved.

So to answer your question, the changes ensured that we were able to surface some specific information we wanted, and could handle certain location use cases.

Beyond that, I haven't noticed any degradation in performance, nor have I really seen any significant improvements.

Let's move on to Ribbon's APIs. How has the experience been building on top of them?

The API is pretty easy to use. It is very well documented and we were able to get up to speed rather quickly. Once we had the Ribbon instance live, there was never a blocker for us to consume the data.

There were some hiccups in terms of Ribbon's ingestion of the data we sent that affected production at times. But we haven't had those issues since we told them to ingest our data into the test environment first. Then, when we get an okay, we can go to production.

I believe there were two or three times where these issues affected us on prod – to the extent that our network directory was not functioning. I would say this occurred two to three times for a few hours each.

So in terms of building on their APIs it was just fine. But for some reason sometimes there were hiccups on their side that caused our instance to not surface data properly, which had to be fixed. That's probably more of a reliability issue than an ease of build issue.

Am I right in understanding there were some initial issues. But once you changed to having them ingest the data into test first before pushing to prod, the issues resolved?

That's right.

The other thing related to building that's a little more technical is they allow us to create custom filters. We can create a custom filter based on one of the flags or a combination of the flags that we've implemented with them. That allows us to carve out certain parts of the network based on the specific plan that a member has.

One downside of working with the Ribbon API is the specialties endpoint could be improved. One of the things that we allow in our UI is to search for specialties. When a member types in something, it queries the Ribbon specialties endpoint to determine which specialties come up as a match. However they don't surface how many providers in our instance are attached to that specialty, which makes it difficult to surface the best data to our members.

For example, there were moments where we had the top four specialties (by string matching) showing only one or two providers, which made our network seem much smaller than it is. One of the downsides of the Ribbon specialties is that they spread them out very thinly, which can be good if you're looking for a very specific specialty, but not as good when you're searching for something like primary care.

Take cardiology, for example. They might split cardiology into six sub-specialties, and then spread the cardiologists across them. Let's say you had 100 cardiologists. Their data may make it seem like you had five in one sub-specialty and then five in another, and five in another, and then 30 in the one that you don't show.

And based on the way their API is set up, it would slow down our network directory product, because we'd have to make a bunch of extra API calls to get that information if we wanted to surface all the sub-specialties. And so we currently haven't done this. That's one of the downsides of the way Ribbon stores their data and how they surface it.

Because a lot of their customers have run into this issue, Ribbon has a spreadsheet that shows how they would group these sub-specialties into a broader specialty grouping. So for example, if you take cardiology, they'll give you a spreadsheet of the UUIDs of their sub-specialties that they would consider falling into cardiology. This enables you to create these broader specialties and potentially surface them at the top of your UI, to reduce the effect of making your network seem thinner than it is.

To play that back: there's no aggregation built into the platform. It's not a hierarchical system where there's a general specialty, like cardiology, and then a bunch of associated sub-specialties. And so you aren't able to pull the data at either the sub-specialty level or broader specialty level.

That's right.

I didn't understand the spreadsheet approach you described and how that differs from just running a bunch of API calls against the different subspecialties.

Yeah, it does not differ from running the calls. Rather, instead of having to run those additional calls, we can still use one call to check which specialties match and then we parse that data to determine if this is a grouping specialty. And if so, we surface it at the top.

Shifting gears, can you tell me about your procurement process? Did you evaluate Ribbon against other competitors in the space?

To my knowledge, I don't think we evaluated Ribbon against any other competitors. I think it was always a "Ribbon versus build it ourselves" kind of decision. And it was pretty clear that at the stage we were at we should just buy instead of build

Were you able to evaluate the data quality or the API in advance?

Yes, using their test UI and instance, as well as by reading their API. We did not have a process where we sent them some sample data and then we looked at the quality of the data they gave back as a result. Rather it was more of a situation where we asked if they could do the things we needed and they could.

That reminds me, part of the reason we decided to buy rather than build was that one of our teammates used to work at a company that used Ribbon. Based on his operations knowledge, he recommended we buy rather than build, and because of his experience working with Ribbon that's probably why we didn't evaluate other competitors.

Got it. You said you didn't go through a process where you sent them some data to see what came back. Was that because it wasn't a possibility or because you got what you needed via reading their API docs and being able to play around in their test environment?

Yeah, it was more of the latter. Mainly because we knew that we had the data that we were going to give to them. So there wasn't really a concern about "are you going to give us enough information back?" In the sense that the size of the directory didn't depend on Ribbon. It was purely that they were doing a cleaning and surfacing step for us. The source of truth was with us – or more specifically our wrap network. That's why when we made the decision it was more about ensuring the API gave us what we needed based on what we wanted to surface. They

also gave us a test API key so we could confirm that on our side and build a sample UI page if we wanted to.

Got it. Anything else to say about the sales process?

The only other thing I'd say is that when we inquired about implementing the custom flags that we wanted surfaced back to us, they were very friendly and allowed us to add them without changing the scope of the contract or adding any additional charges.

How was the onboarding and set up process with Ribbon?

Initially there were a lot of calls. Once things settled down, we had access to our API key, and we had confirmed how data should be sent, we transitioned to a weekly call. This lasted for about four months post implementation. It gave us the opportunity to surface whatever hiccups were arising.

Over those four months, it also became clear who was responsible for our Ribbon instance, specifically who was the main data scientist we could work with. Since then, the support has been really good, particularly during the two or three times where we did have issues that affected production.

Overall I would say that their support is pretty fast and pretty easy to work with. I'm not sure if that's a product of the specific data scientist we work with or Ribbon as a whole.

Let's wrap up with some higher level questions. Overall, what would you say you like most about the product?

What I like most about the product is I don't have to think about whether the data is going to be there for me to use. It just frees up a lot of headspace for me to think about things that are more specific to our business than things that are specific to some data that we're not in control of.

What do you dislike most about the product?

What I dislike the most is the way certain data is not available or I have to sacrifice a lot of speed to get that data. Mainly the specialties example I gave earlier.

I also think they could improve how they categorize specialties, and be more member oriented with their approach. I think the angle they've taken is a very technical one, which makes sense. But to a member, it's not the most friendly.

Based on what you know today, do you anticipate continuing to use the product for the next year or two?

Yes, we do anticipate continuing to use the product for the next couple of years.

Is there anything that would cause you to build it yourself or cause you to go back to the market to look for other solutions?

I think cost would be the biggest driver for us if we were to switch. Or if some other competitor had way better data richness. For example, if they had specialties that are hierarchical and a similar ease of use. If everything was the same, price would be one factor and data richness would be the other factor for us to switch. Those would be the considerations around buying another solution.

Build is probably not on the roadmap. It's not where we want to spend our development resources.

Is there anything that we didn't cover about Ribbon that you think is worth mentioning?

There's a lot that we don't use in their API. I can see a category in their API called Focus Areas where they indicate what a provider treats. There's also an Organizations endpoint, Insurance endpoint, Cost Estimate endpoint, and Price Transparency endpoint – we don't use any of these. So I would just say we use a bare bones version of the Ribbon product, and these other aspects of the product would probably be worth investigating for someone considering Ribbon.

The only other thing I'd note is that I think they're in the process of moving customers over to dropping files through SFTP. This would allow companies to use Ribbon more asynchronously instead of having to send data manually. Since we implemented Ribbon, we've manually sent every data load, which hasn't been a problem for us. But that might be something that someone really cares about when looking at Ribbon – for example, "I don't want to have to think about when I give you a file. I want my server to be able to drop it to you automatically."

Returning to pricing, are there any details you can share?

We did a two year contract. Which I think is pretty flexible. We ended up doing two years with flexibility for expansion.

In terms of structure, it's just service fees based on the number of providers. We pay $X per month for any number of providers up to a cap.