Please stand by for captioning.

>> MODERATOR: Good afternoon, ladies and gentlemen. Welcome to Transparency in Coverage. The webinar will start shortly. Thank you.

Recording in progress.

>> MARVELYN DAVIS: Good afternoon, ladies and gentlemen, welcome to the Transparency in Coverage webinar. Before we begin, I have a few announcements. If you are a member of the press, you may listen in, but please refrain from asking questions during the webinar. Members of the press can e-mail press@CMS.hhs.gov. For those that need closed captions, instructions and the link are located in the chat box.

I will now turn the call over to Elisa Dines in the Consumer Support Group of CCIIO Elisa, take it away.

>> Thanks, on behalf of CCIIO thank you for attending our fourth webinar on the Transparency in Coverage final rule machine readable file requirements. And happy new year. As a reminder the goal of these webinars is to drive the development of machine readable files to a finalized version 1.0. CMS recognizes that the representation of information required from the Transparency in Coverage rules differ from plan to plan and implementations can lead to various challenges such as file concerns and how to represent information in unique situations.

We hope to iterate on the development of information format or schemea of the required data to optimize a consistent file structure. We'll be answering questions related to today's presentation at the end of the webinar. You may ask a question by typing it into the Q&A function at the bottom of the screen. We will do our best to get to as many questions as possible today. And if there are folks that are unable to join the webinar today, we'll be posting the recording of this webinar to our Transparency in Coverage web page.

And the web page is located in the chat box, if my colleagues can add it. A reminder this call is being recorded. So the recording will be available.

So there have been some great discussions on GitHub and questions asked in our previous webinars and over e-mail and to ensure everyone has access to all the responses from CMS we have compiled these, what we're calling technical clarifications, and posted them online, easily for everyone to find. So those are on the Transparency in Coverage web page on the Resources page of that website. And we will add that also, that link to the chat. A link to the website in the chat, excuse me.

And now I'm going to turn it over to CMS' technical advisor Scott Haselton, to delve into what's been happening on GitHub. Scott?

>> SCOTT HASELTON: Thanks, Elisa. So, today we are going to go ahead and review a quick recap of what we covered in the third webinar, and we had some slides, and I'll run through those pretty quickly. I don't want them to dominate too much of the time of some of the changes that we've introduced since the previous webinar.

But, a quick high overview, high-level overview of what was covered in the previous webinar. That provided some of the foundation in the iterative development for some of the changes that we're going to go ahead and cover for this webinar. So it's good to kind of provide that context, as well.

So, last webinar we talked about having the need to have a standardized file name, or standardizing the actual file name of what we're going to be calling it. This will rise mainly through the complexity of having multiple plans, allowing for multiple that we're solving for, is what are the naming standards now for those multiple plans. And that introduced the concept or the idea of the table of contents of which, again, like I alluded to, we're going to go ahead and be iterating some development. Some of which of those changes will be covered today.

So, last webinar we talked about file naming standards for this table of contents, and then introducing this concept called internal reference, provider referencing, which was a great -- a great addition in terms of saving significant amount of file size. But again, providing really what it's doing is providing that foundational, that foundational base that we could continue to iterate on, for some changes that were -- that are going to be introduced quite soon. You could already see those changes on GitHub in a poll request and we'll be covering some of those changes here today on this webinar.

So, moving through the actual, what the table of contents, and the problem that arose by allowing for multiple plans to be -- to be reported in a single file, and some of the solutions and the thoughts around how we landed on our solution. This was typically the standard way of what a file should be named. If it was just a single plan or an issue -- a single plan for a payer or issuer we had the date, the payer's name, the plan name, the type of file, and then whatever the file extension was. So, of course, if CMS was, maybe an in-network file for Medicare, it would look along the lines of something like the date, CMS, Medicare, in-network rates.

And we had that plan data and some of the metadata, like the plan name, the ID and the market type and such. That data was actually on the root node. That information was on the root node for both the in-network and the allowed amounts. And it worked well. It worked well if you're going to be reporting a single plan. And the name, and the name worked, if there was going to be a single plan that was going to be reporting the negotiated rates. So this is some of that data, and we introduced this concept of reporting plans where now you could have multiple plans that could be reported for the same negotiated information. But while this was an incredible addition in terms of a reduction of file size, and duplicative files being reported, it introduced a little bit of complexity when it came to file naming. And this is what we were really trying to solve as a first step. And then iterating on this concept, as well.

So, here's the problem, the heart of the problem in the file naming, is the plan name within the file itself, no longer -- it no longer holds true that there's a single name and a possible solution could be, well we just append every name, every plan name that's a part of the file, and that potentially could look like this. But, in the -- that doesn't really scale in the case of potentially, 50 plans being reported for a single file. So we had to solve for this problem.

And that problem was solved by introducing the table of contents file. Again, this file was introduced to solve file naming standardizations. But then also it's going to provide a fair bit of flexibility for future development, as well. So, the information contained within the table of contents is nothing really nothing new in terms of the information that was being reported in both the in-network and allowable files. It was just shifting that information over into a new file. As you can see here. And then, we have this concept of in-network file object, which really has a location of where you could find the in-network file for the plans that are being reported. Against that file. And that file effectively would just point back to the negotiated information that was done in the original file.

So what this did, right, it would just be that information. So what this did, effectively, was it allowed for the in-network files that are being defined within the table of contents, they could be named anything. There doesn't need to be a standardized name there. Shifted the burden of naming standards over to the table of contents. And not to the actual individual files. That was the impetus of why this change was introduced. And this is optional. If you -- if your organization is working well, which is reporting a single plan per file, that's fine. There's nothing that says that using a table of contents is required. Allowing for multiple plans to be recorded in a file, that option, you're going to have to, if you choose to go that route, you are going to have to go about a table of contents. So we do have standards around -- standards around naming.

So in this case, the in-network file is actually pointing over to this file, and I think the thing that is really, that we’re trying to highlight is you can name this file whatever you want.

So we have our new table of contents file. All of this can be found in GitHub under the examples, both under the examples folder and if you're looking for a more formal definition, the schema folder has the table of contents definition as well.

The new naming format, the thing we were trying to solve for, we can do that now by having a date, a payer's name, and then underscore index, and then the file extension. So in this case, this is what that new standard would be.

And then we talked a little bit on, in the third webinar, around some file size reduction recommendations. Different strategies that could be used. We're going to go ahead and continue some of those conversations today, based on some of the changes that have been introduced. And by the end of this webinar, and then the immersion of the poll request that's open right now, much of the file size concern, or issues that we've been seeing with various organizations, most of those should be fairly addressed now, and we can start focusing some of our development efforts on things that are not related necessarily to file size. So that's pretty exciting.

Of course, there's always going to be iterations that could be made that more or less nibble around the edges in terms of file optimization, file size optimization. But I think really the big pieces are in place in terms of those that were having file size concerns. If you followed some of these recommendation s, hopefully you will be landing in a pretty good place.

So, again, we have these slides up on the CMS Transparency in Coverage website under webinars, if you're interested, or if you want to go a little bit slower than I'm going to go right now. Mainly because we covered this last webinar. But there were five various opportunities that really helped decrease the size of your file. Depending on your organization and how things are set up, you might get different mileage.

So with provider groups, initially, again, I'm going to go through this pretty quickly. Initially we were -- we had this concept of a single provider group with a single TIN that would be associated with the negotiated rate. We changed that to allow for provider groups, being grouped together. So networks or multiple provider groups could be associated with a single rate. Good optimization there. We were encountering some various scenarios there might be a TIN that's the same but there might be different NPIs associated with that same TIN. You might want to approach each group where you're going to be ordering by tax ID number. So the idea here, you just combine these two provider arrays, while still just using the same tax ID number.

So, yes, try to move everything into a single provider group or single provider network, and it should cut down on some of that sizing.

The service code was initially introduced as just a straight string. And we heard feedback pretty early on that the state the negotiated object or price can span across multiple service codes so it was a natural next step to allow for multiple service codes to be reported for a single negotiated rate. So we moved from this string over to multiple strings being reported. So group everything, group all your service codes together to really cut down on how many times that you need to report the negotiated price. If there are multiple -- if there are multiple prices.

We covered this a little bit with the introduction of the table of contents file. This is fantastic in terms of many plans having the same negotiated rates across all items and services. For the same providers. It makes no sense to duplicate that file and just put a different name on it. So, like we mentioned previously, with the table of contents, we allow for the reporting of the multiple plans together. And all you have to do is, in your in-network file object, point to where the network file actually resides. And you will save quite a bit of space for every plan that has that duplicative negotiated rates. Those duplicative negotiated rates. So yes, this will actually decrease the total footprint size of the files that are going to be needed, along with the storage, as well.

Provider referenced groups. We're going to be talking a little bit about this one today. But, to provide a little bit of context on how we ended up where we ended up, it was good to set this initial foundation that we could then iterate on. So, the idea is, as it was previously, each provider group had to be reported with each negotiated rate for each item in service. And that could lead to quite a bit of duplicity in terms of the same provider groups being reported over and over. And there are the potential of some of these larger national networks that could have, you know, hundreds or thousands or tens of thousands of providers. So that, alone, can be large in terms of actual byte size.

But when you start having to repeat that information over and over, that's where you see that multiple of byte size really get out of hand. And it would be great if you could define the network once, and then reference that network within your files. And that was -- that was what we were thinking when this was introduced. So, very quickly, again, in the negotiated rates, you would have to define what groups are part of this negotiated rate, and then the all negotiated price that those groups are associated with.

So what this did was, this is within the in-net work objects. So this is right in the middle of the file. We basically take those provider groups themselves, and then we push them over to a new attribute that's actually on the note called provider references. And this is the same information, and there is one small note here, we have, this is the provider group is just a straight object. In the introduction, a change to the schema was made last week that changed this from an object to an array of objects so it would look exactly the same as provider groups per rate. Just something to keep note of.

But these provider groups would be moved over here, into the provider references object, and then it was up to the creator of the file to assign a unique ID to that provider group. So you want to think of this as almost a primary key, that unique ID that would be associated with the group that's being defined. And then that ID would then be referenced, hence the name provider referencing, within the actual negotiated rates object.

So you define your provider group once, and then you reference the ID of that group X amount of times, depending on, you know, what your file looks like. So, these would be whoever is, like I said, whoever is creating the file, these would have to be managed by the producer of the file to ensure that these IDs are unique, and that you can then associate them.

So this is provider references at the root node, and this is the provider rates. This is what taking this change and then implementing it would look like. Some here, you have your negotiated price. You're effectively just replacing those provider groups with the provider references, with the IDs of the referencing back to, one and two. And then that, as you can see, cuts down on a lot of this provider group noise. Makes it -- the negotiated rate is a lot more streamlined. A lot smaller. And the groups would be associated there.

So, we will build on this idea today. But again, just setting the foundational framework to allow for some of the development that we're going to be looking at today was important in taking the small iterative steps. Hopefully signaled and provided somewhat of a direction of where future development would be leading.

And then lastly, talking about file compression. It would probably be wise to zip up or tar your files. There are a lot of different compression formats to be used. I would just, suppose as a rule of thumb, don't try to venture into any compression algorithms that are super exotic or anything like that. I would stick with the tried and true. 7z, zip, tar, those are all very good and fairly ubiquitous across computing. So I would recommend choosing some common compression format.

And then today's discussion was -- is really going to look back at some of the work that was done in webinar three, and you'll see a lot of the same code. But just leveraged a little bit differently. And that's with external references. And then we've introduced billing modifiers. Which are pretty important. We've heard pretty early on that modifiers such as on a cbt code may have a different negotiated rate than whatever the base billing code was. And it's important that we capture that information versus trying to come up with like an average across all the modifiers. And ending up with a number that doesn't really reflect anything.

So diving into external references. We introduced two different types of references. The external provider networks, and other external valid in-network files. And like I said, both of them are going to be building on the concepts of the internal provider reference, and then the table of contents.

So looking at external provider networks. This is really to help solve some of those questions around like large, national networks, maybe or a plan not having all the information on a provider's network. It's really meant to solve that issue. So through the iterative development, this is -- much of this is just covering what we just covered in webinar three. So I'll move quickly. But we remember, as a reminder we had that provider group, provider network, definition within the negotiated rate initially. We moved that out to a provider reference. By the way, this is all optional. You -- if your plan still would wish to report as provider groups within the ------ rate, perfectly fine to do. Nothing stops you from doing that. These are -- these are file size optimizations that are optional that you can choose to use if you're finding that the file that you're producing is starting to get too large for your needs.

So we took the provider group, we created the provider references, we gave them an ID and just moved the group definition over to the provider references object. Gave the group an ID, and then this is how you would refer to it, the negotiated rate now. Instead of having provider groups, you'd have to have provider references. The schema does define, you need to have either a provider group or a provider reference. You cannot have neither. And at the same time, you could have both if you really wanted to. That could be a situation where there is a large, national network where you would rather refer to with a reference, and then there are smaller groups that you could just do in line with the actual negotiated rate.

And this whole box now becomes the size of this. So we see the reduction of size there. And then here the next steps from where we left off last week is we had this icon ID and the provider references, this is now going to be moved within the provider references we're going to be allowing for the reporting of a provider group or we're going to have this concept of a location, which would be a URL that would point to a provider network, object. Which will -- we'll go ahead and show you in a little bit here. But the idea is, is you're taking this provider group(s), which has been bouncing around, not actually changing, but changing locations, not changing structure, but changing locations on where it could be referred to. So taking these groups, and moving it into a location that could be referred to externally, actually solves some of these problems that we talked about earlier with large national networks or unknown rented networks.

The ID is still needing to be reported because that is the reference point for the negotiated rate for the external file. So, this provider group ID will still need to be managed by those that are actually building out the in-network files so that's important to keep in mind.

And now this is what a provider file would effectively look like. It's that provider groups, those provider groups were moved from the negotiated rate objects to the provider reference objects, and now effectively culled out and put into its own JSON file, again just want to stress that the structure of this object or this array has not changed. Its location is changing, and providing options on how you would like to reference that structure is really, really what we're looking for here and trying to be flexible based on the diversity of plans. The diversity of plans that are out there, with the unique constraints that they have. Yes that provider group is effectively moved into its own file.

And there are three things, if you want to leverage this, there are three things that need to be done. The first is make sure your provider reference is actually defined. So I'm going to be jumping into an example here. This is the in-network example. Here I have provider references being used in a couple of ways. I defined some provider groups. But then I also have the external reference. So if you're going to use this, you must define where the location of that provider file is. There are some restrictions in the schema that requires the URL to actually be https. We have documentation in our readme that says all of these files need to be served over https. So you'll -- you'll run into some validation issues if you don't have a fully qualified domain with https. But this is number one, you need to have a location.

Number two is, you need to actually have the file itself, this is what a -- this is an example of what one of those provider files could look like. In this case this provider file just has groups of networks. Provider networks. No IDs. You'll notice there's no IDs here. It just has the information that -- of the provider network that could be then referenced. So you're going to need this file, the actual file to be a valid provider reference file. And there was a new schema that was put together just for this. But if you look closely you'll see that the schema actually, to validate this file, hasn't -- it was just pulled from the existing schema, because this provider group object hasn't changed in its structure. Just wanted -- I'm really hammering home that we haven't been actually trying to change these core structures too much. Just where they're located is really the optimizations that we're looking for. You need the file, the provider file itself. And it's going to look like, something like this.

And then lastly, you're going to need the actual ID. This is the reference ID being used in the negotiated rate. But if you're going to leverage this strategy, you need a way to really manage these provider group IDs. And what that will look like. So those are the three -- the three things that are needed, if you're going to be using external provider networks. You're going to have to go through each three of these every time.

That said, I think it is a pretty small -- the complexity that's added is relatively minor compared to the benefits that would be gained in terms of, again, central file size savings. This is actually pretty exciting. Because we have been hearing, very legitimate concerns, around some of these national networks being massive. And to repeat that network within a single file X amount of times for every item in service was going to be a hard hurdle to overcome. Providing that file -- providing that file once and then -- or providing that network, that national network once, and then referencing it over and over was a great win, as well. But at the same time there were times where plans may not even know the network. That's associated with these negotiated rates.

So, then, allowing these plans to point to external networks that are managed by another company was really, really the thing that we were hoping to solve there. And at the same time, save on file size. So we'll be curious for feedback, on some of the feedback when it goes through implementation. To see if this is working well, or if we need to continue to iterate on this. But for the time being, provider groups, provider networks and referencing them within the negotiated rates, that development is nearing finalization. It's not finalized.

And then referencing other valid in network files. So this builds off of that new table of contents file that we talked about. And this is also really exciting, and hopefully will cut down on some of the file size concerns that you've been running into, that are very legitimate that everyone's been -- a lot of people have been running into.

Here really focusing on the in-network file object that was introduced with the table of contents file, again, as a reminder, just talked about it 15 minutes ago, but this table of contents allows for multiple plans to report on a single file. Then this in-network file object is the location of where that file would live.

So this is kind of the successive changes that we -- that were introduced to this particular object in this file. The first thing that we did was we created this, and we changed it into an array. And then that allows for multiple in-network files to be defined that would be -- that would be in aggregate a single in-network file.

And here, this is why or how you might use this. You may have your in-network file with all of your -- all of your plan's negotiated information. And this would be your core file. But there are negotiations where there are items and services and networks that are shared, rented, and referencing those files or trying to pull in that information, then depicted to your file, could be challenging.

So, we are allowing for multiple files to be combined together within the table of contents file. And here's an example of, you might have saw this in some of the discussions, this mirrors pretty closely to what was being talked about. But we have now our in-network file along with this behavioral health network shared file. This really could be anything. Any in-network file. It allows, again, for the ability to define things once, and then just reference it, rather than having to pull it in from various locations to build your -- to build your ultimate file. So convenient, as well.

Things to keep in mind, though. If you are going to be using external in-network files or referencing external in-network files, is the file that is in external that you're referencing, it must be a valid in-network file. It needs to comply with the actual in-network schema. It can't be like a subsection of, you know, the in-network rates object or something like that. In totality that file that you're referencing needs to be a valid in-network file. So keep that in mind. If you're looking to employ that strategy when building out your files.

And then lastly, today, want to cover billing code modifiers. The actual change itself was fairly minor. But, I think it supported, the change itself, it was minor, but it addressed some concerns around how negotiated rates are determined, based on the billing code. Like, mentioned earlier, the base rate for a billing code may be completely different. The negotiation rate may be completely different if it has a modifier on it. So, what was to be reported? Do we report whatever the base code is? Do we report only the modifiers? Do we do an average? And this introduction is really a clarification, and from a technical standpoint, not really a heavy lift to introduce. And it provides quite a bit of clarity on what's actually being reported.

This is already -- this was in a poll request last week, and it was merged fairly quickly into development, because like I mentioned, the development or the technical development behind it was actually not too complex or difficult.

But running through what that actually is, we could go and take a look at the pull request details, if you're interested. And it really is, I didn't look at this pull request. The introduction of the modifier for the allowed amounts is actually within the allowed amounts objects. So, if you remember, out-of-network object that really has kind of the core of what your allowed amounts are going to be reported. And then within that object there are -- you have your concept tin, your service code billing class. And we're adding the billing modifier there. And it's going to be an array. And you could have as many modifiers as you want that are tied to that single item that's being reported on for the allow ed amount. Initially this was going to be a string. But, every time that we put a string, and thinking that, you know, modifiers are going to be unique, in terms of their billing, we find that, you know, every plan -- no two plans do negotiation or reporting the same way. So, we went ahead and just made it an array to begin with. Without actually knowing if the array will be leveraged too significantly. And I'm sure, as data -- as reporting will be required on July 1st, we'll get some data to help inform whether keeping on an array makes sense, or changing that to a string.

But this is the location of where you're going to find it in the allowed amounts. And then within the in-network file you're going to find it in the negotiated price object. I copied this anyway. Okay, no problem.

So we'll go to schema's, the in-network rates. And then the modifier. And this should be in the negotiated price object field. So we have the billing class, the service code, the expiration date. This is where most of the core negotiated information is going to be reported. And then we've included the billing code modifier. Again, it is an array. No, it's not required. And if it helps you with your reporting, of various negotiated rates that are modifier dependent, then please go ahead and leverage this, and this is effectively what an implementation would look like.

Hopefully this is viewable. We have our in-network object with an item or service being done. The negotiated rates for that item or service and here are the provider networks that we talked about earlier. And then the negotiated price itself. And then we have some information that we covered in previous webinars. With the addition of the billing code modifier. And we're going to be placing it within the negotiated prices. Mainly because, by introducing a modifier, it may create another object. And rather than repeating the whole negotiated rates for the in-network object one more time, by putting it down, let's say at this level, keep elevating it to the negotiated price object, should save some space. And makes the most sense there.

Again, it is an array of strings. I'm sure of how much mileage will be gained by having an array. But we're going to go ahead and go with it until we get more information. Then, that is -- those were the two introductions that we have for you for this webinar.

Just a reminder, the -- those were the modifier, and then the external references. Hopefully, we have been showing you over the course of these webinars really what our development strategy is here, is to accept some of those foundational steps. And iterate in smaller pieces, and making sure that those iterations that are being made and those changes that are being made, really making sure that the file itself, as it was, you could still produce that, and that's a valid file. These changes are optional.

You don't have to employ these strategies if you don't wish. And you can report -- you can report your file in the way that was defined before these webinars started. That's not a problem, either. So with that said, before -- before moving on too far, we're going to have a polling question --

So like I was mentioning earlier, a lot of these additions and updates were really targeting file size, file size optimizations. And we have the larger pieces in place. And like I said, you can nibble around the edges on other optimizations but the large pieces are in place. And really looking forward to some of the next sets of difficulties, problems and clarifications that are -- that are with these machine readable files. We're going to be really turning our eye on providing some guide ABC and some implementation guidance around some of these alternative non-fee for service arrangements. And you should really be seeing, with the coming iterations and the coming changes, really the focus will be on that front. Unless new file size concerns are brought up, or there was some other optimization that could be employed that would be an iterative change on where we are right now.

But if not we're really going to be concentrating on providing support for some of these alternative -- alternative methods. And, with that we've been hearing quite a bit of percent of build charges being a method of some of these contractual methods we set up in such a way and we would really like to get an idea by putting out a poll of how often is your plan running into negotiations that are based on some of these percent of billed charges. How prevalent that is. We will be certainly addressing it, but just want to get an idea from the community on how often you're coming across this scenario. So I'm going to go ahead and launch this poll. I'll give it a minute, and then we can move on to some questions.

>> I'm just taking a look at these poll results. If you haven't yet, please pick a percentage. But yeah, based on GitHub discussion, we understand that some plans need structure as percent of charges. And I know folks are asking about disclosure in that circumstance. So just want to assure everyone that the departments are considering what we're reporting for the in-network negotiated rate file will look like in these instances. And for some of the other instances where plans and issues were unable to derive the dollar amount to report. And we plan to get guidance out to the community soon. But we are really encouraged to see that GitHub is functioning as independent in the final rule. Getting good feedback. Hopefully some useful feedback and working towards a schema that makes sense.

And as we said in the final rules, GitHub is our preferred method for providing technical assistance to plans around specific use cases. Such as alternative arrangements. So really appreciate your participation in that form. He can now, I think, since Scott's ready, we can move on to addressing some of the questions that came in through the Q&A feature.

>> SCOTT HASELTON: Yes. Thank you. Starting -- thank you for everyone that answered the poll. Again, it's just invaluable getting that feedback to understand where some of the concern is. So we can make sure that we're focusing our efforts on some of those larger issues. Looking through -- looking through questions, this is always -- this is always a challenge to read some of these so quickly. Can the time line expectations be discussed for poll requests that move to the developed branch? How long will these changes be developed before publishing to the master branch? The question really is, is -- the question was straightforward. It was asking, stands up on its own, but the idea is, when new changes are introduced, what typically happens is they're done on something called a new branch that is not the main branch, and then they go through a poll request, which allows the -- allows the community to give input, and look at what changes are coming down the pike. And it goes through a series of branches. Typically, in this case, this branch is named external references. This branch will be pulled into the develop branch, which is supposed to be a more stable branch of changes. And then lastly, that develop branch will then be -- that develop branch will then be merged, or there will be a poll request that is made into master. And then that's effectively, when it's a master, it's effectively minted, meaning that it's -- that is what -- that is what compliance will be based on, so with smaller changes, it can move through these branches a lot quicker. I think with the modifier, there was a branch with the previous changes, with let's say for example for the modifier code, the change itself, as we saw, was fairly small.

And that allowed for -- I mean there wasn't going to be too much back and forth on the technical merit of where that code was being introduced. So it can move into the developed branch pretty quickly. Larger changes such as this external reference branch is a pretty large concept in getting input on how to -- if it makes sense on how these concepts are being put together. And really trying to capture that as far back in this development process as possible, really saves time. If we were to push this external reference into develop and then into master, at a quick clip and then we find out that we need to change something, or there's some core concept that, you know, wasn't considered that's actually a better implementation.

So it really depends. And I understand it's not a satisfactory answer but it really depends on like the complexity of what the changes, and how it's being introduced. I would imagine that after this external reference is into development, there hasn't been a push from development in the master in some time. And like I alluded to, much of these -- much of these changes over the past couple of webinars have really been focused on file sizing. And I feel like, you know, we could cut -- we could cut a release that basically addresses a lot of those file size questions from develop into master after this has been merged in.

So, that's more or less the time line expectation. We have other pricing determinants similar to modifier, such as age range, gender provider specialty, are there plans to add other pricing determinants to the schema? This one is interesting. The schema itself is meant to be agnostic of beneficiary -- beneficiary modifiers. This one might have to take back to get a little bit more information. If you'll remember the final role itself is comprised of two things. One is a tool that would be provided by -- a tool provided plans that would take it in beneficiary information to provide an estimate for I think it's 500 items of services. And then the second was the machine readable files which are mainly supposed to be agnostic or don't really take in the beneficiary information into consideration.

But, we will have to get back to you on that one and see what those actual contractual arrangements look like. --- would be that you would ---- recording whatever the base rate is without having that age range, or anything you'd like beneficiary specific. But we'll have to provide some clarification on that.

Here's a little bit of an extension on the pull request time line for example, some poll requests. Having closed but only develop branch -- trying to understand -- okay which is the official schema, it is, master is the official schema. Like I said, trying to solve most of the file size issues and I feel like we were pretty close there. So we could expect the develop branch to move to the master fairly shortly.

For the reference to external files, when you have a rental network there will be some duplicated providers between the local file, and the rented network files. For example, the local service area you may see the same provider in both files. The machine readable file consumer can de-duplicate the provider. Is that acceptable to refer to the large -- yeah, yeah this is definitely some complexity that was considered when introducing the external files. There may be a bit of duplication based on what in network providers are a part of. They could be a part of multiple networks. So, that is very much a possibility, and that's kind of to be expected. And that's okay. Ideally, I mean you want to try to de-duplicate as much as possible. But there's no hard requirement that says that they absolutely must be de-duplicated. Because a lot of these networks are not going to know what providers that are a part of other networks. It's just they don't know that.

So that's acceptable. It's okay to have those duplicates across the reference -- across the external files.

For the provider network group slide, the introduction of location. Let me make sure that we go back there to see. Location right here. Okay. Is the uniqueness of the ID only applicable to the file it points? Does a provider group ID still need to be unique in the file or just unique to the file location? No, it needs to be unique to the provider references array. And I think the best way to answer this is maybe show you an example. Under the develop branch I'm looking at the in-network rates fee for service single plan sample. Here there's an example of provider references. And the provider group ID needs to be unique across the different provider reference objects.

So here's one object. And in this one object we have two networks that are defined in the file itself. And then the developer of the file needs to give those provider groups ID. And then, on top of it, there may be oh, my gosh, I'm sorry. I'm showing you the wrong -- again, same thing. There's two networks. The two provider networks within this one reference, and the developer must show the ID. Or define the ID. Here we're going to go ahead and add another reference. This reference itself is not going to be defined in line, or in the file itself. It's going to point to some other rented network, or some other provider group. And then all I'm doing is just incrementing one and then the next one is two. It's up to the developer to manage these IDs and make them unique to other IDs that are within the reference array. The provider references array. And these IDs will be associated to the networks, the provider groups to the networks or the location in which they're grouped together within the object.

The introduction of the table of contents in version 0.8.0 is optional, but can you confirm that if we need to use the table of contents if a single file will be generated for multiple plans? I'm not -- so, maybe this is a question of you want to have multiple plans be grouped together for a single file. If you want to go down that option, rather than having each plan report that file with the same negotiated rates, right now the base requirement is one plan for file. If you want to go down the option of having multiple plans per file, again this is an option to you, then, you are going to have to have a table of contents file. That's what is meant by optional. You don't have to report multiple items together. But if you do wish to do that option then yes you're going to need a table of contents, going to need a table of contents file.

And just a reminder, that table of contents file was created to solve the problem of file naming. And that's the reason why it's there. It's also provided fruitful benefit in terms of future development that we saw today in having a place to have multiple in-network files be referred to for these plans, these plans and issuers in making up their actual file. I just realized I'm actually one minute over Elisa. I think there's a lot of good discussion here. A lot of it is like good guidance, things to chew on. And a lot of it is talking about what some of these alternative payments, or all these alternative contractual situations are, which likely we hope to really focus on in the next upcoming webinars.

So I'll pass it back to you and do my best to go over these questions, and if they're applicable to some technical guidance, I'll do my best to answer. So, thank you.

>> And yeah that's all the time we have today, as that said we'll do our best to get to questions, GitHub raised a lot of interesting issues and we're chewing on them. So, thanks again for all your participation. Our next webinar will be scheduled to be, you know, in about two weeks from now. We'll send out an invitation shortly, and please remember to register either on the transparency and coverage website, or GitHub site. And to have a great day. Thanks again.