>> MARVELYN DAVIS: Welcome, the transparency and coverage webinar will start shortly.

>> MARVELYN DAVIS: Good afternoon, ladies and gentlemen.

Welcome to the transparency and 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. If you have any questions please email press@CMS@HHS.gov. For those that need closed captionsing the instructions are in the chat box of the webinar. We will be asking questions related to the technical implication of the machine readable file requirements today. You may ask a question by typing into the chat box at the bottom of the screen. We will do our best to get as many questions as possible today at the end of the presentation. 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. The web page is located in the chat box. Now, I'll turn this call over to Elissa Dines at the center of consumer information and insurance oversight in the Centers for Medicare & Medicaid Services.

>> ELISSA DINES: Thank you, on behalf of CCIIO, my name is Elissa Dines, the deputy director for consumer advocacy and the consumer support group in CCIIO. I have Scott Hazel ton with me during the development and implementation of the transparency and coverage rule.

>> Recording in progress.

>> ELISSA DINES: He's also the CMS representative in GitHub. So you may have already seen him around. We've instituted these webinars to support plans and issuers' efforts to move to compliance for the transparency and coverage rule in machine readable file requirements and provide an additional platform to address questions from developers on the schema. We intend to hold these webinars every three weeks as needed, and will announce the schedule of future calls and websites on the tick website and the tick GitHub web repository. Post call we'll provide a recording for tick website. We ask that you post questions related to the call discussion on GitHub, and that link is available in the chat and in your registration email.

That way we can thoughtfully address questions during the webinar or later on GitHub please note your question will drive further content so be sure to provide content. CMS recognizes the representation of information required from the transparency and coverage rule can differ from plan to plan and implementations can lead to various challenges, such as file size concerns on behalf of those producing the required files.

Over the next few months and beyond, we hope to iterate on the development of information format of the required data elements to produce a more optimized and consistent file structure.

In the transparency in coverage rules, the department stated their intention to make available non-substantive technical imitation guidance through the collaborative GitHub platform. We believe this is the best way to facilitate technical assistance in addressing how various plan designs can comply with the requirements of the final rules. In the GitHub space, offers the department opportunity to collaborate with industry including regulated entities and third party developers to ensure the file format is adapted for reporting of the required public disclosure data for various plan and contracting models. Plans and issues will be able to access the GitHub schemas at any time and collaborate with the departments in real time. CMS decided to use GitHub as a way to offer an iterative, transparent and communal develop process with a technical audience responsible for gathering the information required in the transparency and coverage rules and production of the machine readable files.

These meetings in GitHub are not meant to address policy questions or non-technical issues that the community might have. These webinars in GitHub are meant for developers to discuss technical implementation. Using these tools is fairly new for CMS but quite common within technical development within industry. CMS wishes to leverage the benefits of iterative development through the GitHub platform through its traditional development policy process with responsible feedback loop for all parties involved. Again, the department's goal in using GitHub is to facilitate this collaborative effort to allow plans and issuers to meet the disclosure requirements of the final rules while addressing I.T. system, issuer and plan attributes. To the extent unique negotiation scenarios are not addressed sufficiently through the technical implementation guidance, the departments intend to help all plans and insurers meet the requirements under the final rules. GitHub is the preferred place to have technical discussions, so please ask questions there. If you prefer or feel like a question is best answered over email, you can always email our inbox. The instructions are available on the TIC website at the link in the chat. We understand traditionally STA static documents such as copy books and dictionaries could be provided in the form of a PDF on the website. They would traditionally be used by stakeholders for implementation.

For implementation of the TIC mean readable requirement, we hope to be more akin to how technical development happens in the industry. -- please be aware those static documents will be driven by the developments happening on GitHub and should not be considered final. So we recommend invoke the final rule and PRA to go to GitHub for the most up to date information.

The technical implementation guidance hosted on GitHub will include a repository set of schemas needed for compliance for the input data rate file and the allowed amount machine readable files. Once schema 1.0 is locked down, we'll post all final materials on the TIC website and on GitHub. Until then materials can be found on GitHub. And to ensure all information is being communicated out from GitHub to a centralized place, the transparency and coverage team will keep a running list of all clarifications made on GitHub.

These coals on the transparency and coverage website on the resources page. We will update the list as frequently as possible. You can find this running list on the resources page on the TIC website and at the link in the chat.

We will also, of course, use these calls as a way to communicate out any widely applicable clarifications made on GitHub. Later in the call, we will also go over how to monitor changes on GitHub so you can see updates or clarifications in realtime.

On today's call we'll provide more detail about how the departments are collaborating on GitHub and for those unfamiliar, provide some assistance on how to interact with the site.

Again, please use the Q&A box at the bottom of your screens to submit questions. We'll try to answer as many as we can get to on the call. Before we dive into GitHub, I wanted to remind folks of the issued guidance on August 20th delaying enforcement of the machine readable file requirements for the in network negotiated rate file and -- the charges file by six months. So now issuerers and plans are required to make available to the public those two machine-readable files starting July 1, 2022. This guidance also have the department's deferred enforcement of the requirement to publish machine-readable files related to prescription drug prices pending further rule making. A link is provided on GitHub and the CCIIO website.

I'll turn it over to technical adviser Scott Haselton to dive into what's happening on GitHub.

>> SCOTT HASELTON: Thank you, Elissa. With CMS leveraging GitHub as it it rates the schema, this is a good opportunity to level set on what GitHub is and how it is used.

Today we're going to be covering some of the basics as well as our approach and timing to get to that version 1.0 for the required schemas. But before getting into GitHub, I wanted to cover some of the basics on how CMS plans to communicate updates when they actually happen. We plan to post updates on the transparency and coverage website with technical clarifications made on these calls and webinars through GitHub or any other CMS deliberations. When updates are made in GitHub, CMS will post on the transparency and coverage website, and a link back to get HIB covering the discussions, changes that were had. We plan to communicate out decisions and changes made on GitHub or other CMS deliberations at the start of the call. Any output will be communicated on GitHub to ensure the feedback loop is complete with these various avenues. As Elissa noted, for the most up-to-date information on anything related to the development of the schema, GitHub is the place to go. With that said, I would like to run a quick poll to get a sense of the audience's familiarity on GitHub.

>> SCOTT HASELTON: Thank you.

Sorry for the delay. We'll let that poll run for about a minute while we gather your answers. #o it looks like roughly 50% have either no experience or just have heard of it. I will stop the poll.

So this is a good opportunity to level set on what GitHub is and how we're going to use it.

So first, there are no signups or accounts that are required to actually view the transparency in coverage information on get hub, but to participate in the discussions, you will need a GitHub account. Signing up is straightforward by visiting GitHub.com. Keeping up to date on the latest developments and discussions happening on GitHub is fairly easy. GitHub can email you any time a discussion is updated or when a change to the schema is made to ensure that you're following along with the latest developments. To do that here, we are on CMS's transparency in coverage GitHub implementation repository. What you would do after signing up is go over to the watch button over here, click all activity, and you will be notified by GitHub with the email that you use to actually sign up. CMS is providing both the schema of the required format for the information required in the transparency in coverage rule as well as implementation examples to help answer specific technical questions, and to view those, the home page on the price transparency guide repository looks like this.

You'll have up here what looks like a file system that you can navigate through examples or schemas with the base documentation of what the repository is, some of the background, how to keep up the date, recent guidance and other useful information that you're going to need when developing some of these files.

Navigating the site is fairly akin to how you would navigate a traditional file system. You can click on some of these files, such as examples, and you can see allowed amounts or in-network rate examples. You can click further and see actual example MS by clicking on the files themselves. It may take a minute to get used to it, but it's fairly intuitive once you've tried it a few times.

Another main component that CMS is leveraging is the interaction with the actual developer community. This is happening within discussions which you can find here at the very top.

Discussions within GitHub are meant to be productive insights into different use cases that surround implementation such as various complex negotiated rate scenarios, different allowed amount scenarios or provider groupings. It's really through these discussions that get -- on GitHub that CMS can understand the various rate imitations to help drive the structure of the required files. There have already been fantastic discussions that have let with changes to the schema to allow for a more optimized output of the required information. CMS is eager to continue down this path of iterative development with the continued input from the community. While the structure of the file may continue to evolve with this input, the actual required information that's set forth within the transparency in coverage rules will likely not change. More likely there will be situations where a field will be added to the schema to help contextualize already required information.

Again, CMS will communicate any such changes on GitHub and the transparency in coverage website and on these calls. With that in mind, development of these files can get started right now with the understanding of what the required information is. You can find that required information by going into the schema's folder, picking the file that you wish to develop or get more information on and all the documentation will be found there. The formal schema could be found either through The dot Jason file or the dot XML file.

Over the course of the past few months there have been some changes to the schema to help address some of the file size concerns, the legitimate file size concerns. We continue to look to different optimization suggestions. If you've been on GitHub, you would have already seen some of these changes. But one of the goals in having these webinars is to communicate those updates for those that are actually not engaged in GitHub.

# With that in mind, we'd like to cover three of the more recent updates that help address some of these either file size concerns or different unique implementation scenarios. The first is with the service code attribute. The service code is displayed in the schema of both the in network and allowed amounts files. Before the change the service code attribute, which you could find right here, was located in something called the negotiated rate details objects, along with the tin and providers right here. The tin and providers, an array of providers of NPIs to create something called a composite key or unique identifier for the negotiated rates. The service code was an individual string.

Through GitHub discussions it was quite clear that multiple service code locations had the same negotiated rates and representing that data would lead to a lot of duplicated data. While there were service codes that had the same negotiated rates for some items and services, there were also arrangements where the negotiated rate is dependent on different service codes. This was a great opportunity to make an update that would help address both situations while dramatically driving down the file size with those current requirements. As you can see here, the service code was moved from the negotiated rates objects. We have the removal of the service code here, and it was put into the negotiated price objects. This type was also changed from a single string representing a single service code to an array of strings that would allow for multiple groupings of service codes to a specific negotiated rate here. The goal was to save the duplicity or duplicative negotiated prices that may be associated to the combination of the providers to that tin.

The second major update or guidance update -- this is more along the line in guidance, and will be included in the traditional -- CMS's traditional FAQs on the transparency in coverage website is what to do if an allowed amount doesn't mean the 20-claim threshold.

Quick background, the reporting of data and the allowed amounts files requires more than a 20-claim threshold for each item in service to avoid potential privacy issues. The questions of the reporting was required, if a plan didn't meet that 20-claim threshold for any of its allowed amount had been raised. The short answer is, yes, producing an allowed amount file is still required. An example of what that would look like is here. #The information in the allowed amounts file will be minimal because there isn't much actually to report on. The file, though, still needs to be created so consumers of the file know that plans being reported on do not have any claims that actually met that 20-claim threshold. If plans continue not to meet this threshold, the maintenance of these files will be extremely minimal. An example of an out of network or allowed amounts file that does have out-of-network reporting, just to give you a sense of the difference between those files will look like this. You have your reporting plans and then you have your claims that have met the 20-claim threshold. So this array of objects will be populated, whereas you still have your reporting plan, but that array of objects is just empty.

Lastly, the last set of updates that we want to give here is the ability to report multiple plans inside a single in-network and allowed amounts file. It has become quite clear that there are multiple plans and offerings that have the same sets of negotiated rates for all items and services for the same providers. Initially reporting was targeting a single file per plan, but unfortunately, as the community has brought to us, this leads to a lot of large duplicated files for plans with the same information, with the same negotiated rates. #here was a healthy discussion on GitHub that helped bring the problem to light. As the community discussed the scenario, great solutions were brought up to address the problem, while not introducing substantive data element changes. The recent change looks like this in the schema where we had -- this is for the in-network file where we had initially the plan name, the ID, the type was reported on the base routed of the schema, only allowing for one plan per file.

With the update, you'll see here that CMS introduced an attribute which is an array that allows for multiple plan objects to be included in a single file that have the same negotiated rates for the same providers and in this case this example has Medicare and Medicaid as having those same negotiated rates with the providers. #he updates for both the service code and allowing multiple plans to be required in a single file are great leaps in file size optimization. CMS will continue to make these updates on GitHub.

You can follow along as well as being part of the discussion and solution for future updates.

Of course, these updates will be communicated out in future webinars if you don't wish to monitor the actual on-the-ground activity happening on GitHub.

Next I want to cover how we intend to use versioning of the schemas when these types of changes are introduced. So for developers and consumers of the machine-readable files, to have a clear understanding on what data they are working with or what instructions were used to produce the file, there needs to be a way to track the version of the file. Tracking the current version of the schema will be done through a process called tagging. That will look like this. Here on the home GitHub page we have a link that says tags, and this will allow you to look at a historical updated version as the schema gets updated. Here we have 0.1, 0.2.

These tags are driven off of something called a version which can be found here in the schema documentation. The attribute version is a string, and it tells the number in which the file is produced and ultimately consumed to help clarify the information of what that schema was in that moment of time. #hen changes are made and an update to any schema, the GitHub commit will contain the new version number on the schema. It is the tag and ultimately the version.MD, this is the current version, 0.4.1, that will inform both the developer and the consumer of the files the version of the schema in which the files are being built against. #ach file has a version attribute that will be populated to denote what version of the schema was used in producing the files. So back to what that looks like, we have the tags, and you can click -- these are moments in time in the past. If you wish to see what version of the schema was being produced, you can go back and compare it to a previous version. You can see all the updates that happened between these two versions. It's another way to easily track source control. #o today we wanted to cover CMS's approach to getting to version 1.0 for all the required files.

Like policy, the work of soft wear development is usually never done. There's usually some development happening whether they're bug fixes or introductions of new features or making larger software product changes. Technical development of this schema is no different.

CMS will be following the standard versioning process found in many software development projects with a major, minor and patch number to represent the current version of the schema. CMS plans to release a release and master branch to continue this development. The following is the guiding principles for the version numbers that you can see here.

We have a number dot number dot number. So these three numbers are represented in a major dot minor dot patch number. CMS will consider version 1.0.0 to be the version in which to follow to be considered compliant. A major version -- when you iterate on these numbers, it's good to have an understanding of what number gets bumped when or incremented when. So the first number being the major version number is when an incompatibility changes are introduced beyond version 1.0. A minor version is when attributes or values are introduced or removing in a backwards compatibility manner -- removed in a backwards compatible manner. The patch version, which is the very last, is when backwards compatible bug fixes are introduced. Over the course of the next few months, updates to the structure of the schema will address file size optimizations. Negotiated rate, contextual clarification such as, is it a facility or a non-facility negotiated rate, and others driven by the discussions had in GitHub, and those will trigger a minor number increment in the schema as we have seen here at the schema becomes more mature. The minor is incremented. The combination of timing and the amount of active discussions had within GitHub will prompt CMS to move to a version that is going to be 1.0. Like we said, while CMS recognizes that software is never done, there still needs to be a line that's drawn in the sand that, quote, unquote, finalizes development. If discussions with within GitHub are still active up till March 1st of 2022, CMS plans to move to version 1.0 while allowing for continuation of development.

This would mean there is a possibility for the schema to have a version 1.1, 1.2, et cetera. These would be the last iterations for which developers can produce files against, but would only be required to conform to a version 1.0.

Versions moving beyond 1.0 should be seen as a signal that further iterations of the rule or possible PRA process that is intended in the future. Also, versions that move beyond 1.0 have the possibility of future file size optimization changes.

So it would potentially be in the developers or the producers of these files' best interest and consumer of the files' best interest to try to stay up to date. Ultimately compliance will be on version 1.0. # Before moving on, I would like to get a poll of what some of the larger buckets are of interest to this group, understanding this first webinar is pretty basic in level setting in what GitHub is, how to use it, who U to participate with the next few webinars tackling some of these meatier questions.

# I'll let that run for about a minute.

>> Can I just add, if your selection is other, please feel free to let us know in the Q&A what the other is.

>> SCOTT HASELTON: Thank you, Elissa. #hank you for that feedback. We would also be interested in what a desired cadence for these webinars would be outside of GitHub to help with that communication loop. So I'll be putting a poll up for that as well. #

>> SCOTT HASELTON: Thank you for that feedback as well. #lissa, before we go on to some of the Q&A, can you talk to folks about future webinars?

>> ELISSA DINES: Sure. Thanks, Scott. So we're going to announce future calls once we get a sense of what frequency would be most useful to folks based on GitHub feedback and the poll responses. I just want to remind folks that these calls are primarily to address technical implementation questions. We received a fair number of policy related questions through the Q&A and we won't be able to address those today. To the extent you have policy related questions, feel free to submit those to our mailbox in the TIC website. We carefully consider each policy question. We're also happy to discuss policy related issues on one-on-one calls. But again, these webinars are pretty much focused on your technical implementation questions. I'm also note that even some technical implementation programs are best answered on GitHub, things like questions around specific use cases, optimization recommendations and general development are likely best asked on GitHub to allow for a more thorough conversation and solutioning process. This also allows for others to participate as well.

I also want to emphasize again that we're driving to lock down schema 1.0 by March 1st. And then before I hand it over to Scott to handle some of the more technical questions, I did see a question related to the prescription drug file. Someone flagged the transparency in coverage PRA posted to reg info a few days ago and includes the appendix on prescription drug files. Even though we're deferring enforcement of that requirement, so that's included in the PRA as it is still a requirement in the final rule.

So until we make an actual amendment to the rule, it continues to be part of the burden that CMS that the department estimates for the rules requirement. Don't be thrown off if you see that appendix.

Okay. With that, I think I will hand it back over to Scott to see if there are questions he can answer in the Q&A box.

Scott, what did you find in the Q&A box?

>> SCOTT HASELTON: A lot. Here is one. Is the schema version part of the naming convention so we don't have to read the file to determine the active schema?

At this point, it is not part of the naming convention. If you want to go ahead and start a discussion where it makes sense to put it there, there are active discussions around naming right now. Currently before the multiple plans being allowed and in a single file, the naming convention was to use the date in which the file was produced, the plan and then the extension.

But with the introduction of multiple plans per file, naming is still being determined on what that might look like, and then that ultimately itself could include the version of the schema or the file that was produced to avoid having to actually download and read the file in its entirety to see if the actual version is updated.

So here is a question that we've gotten a few times through GitHub. It's pretty common. So I'm going to take some time to cover it here to make sure everyone is on the same page.

There is a lot of questions around the negotiation on the billing code type, which is --

CPT hit picks DRG. Initially within the rule itself, it gave a list, a non-exhaustive list of common billing code types. There are some standard billing --

types of billing codes, and then there are some that negotiations are made on that are not so common. And in the rules, it actually allows for some flexibility on the billing code in which the negotiation happens. I will put this up in chat. Currently we have this list of running billing codes that are allowed in something called an enumeration for the billing code type attribute.

This list started out pretty small, but grew pretty large within the first few months of the initial release of the schema. If there are billing codes that people are negotiating on that are not included in that table, I would highly recommend a user opening up a discussion to talk about other billing codes that may be used, and we can go ahead and get that added to the table. #Here is a question that's pretty common as well. What are the acceptable formats in producing these files? What file formats?

CMS chose to be fairly non-prescriptive when allowing the producers of the file formats make the file. The boundaries -- really the guiding principles are that the file format itself needs to be non-proprietary and needs to be of an open standards format. So, quickly, what that means is, if you're using like a PDF to build these files or an out put of these files or a word doc or Microsoft excel, they're proprietary formats. It's important that a non-proprietary open standard is used to lessen the burden of those that wish to consume the file so they don't have to buy Excel or don't have to buy a proprietary program to actually read these files.

Of course, each format, when determining what format to use, each format has its strengths and weaknesses, so selecting the right one for your organization is really important. There have been suggestions on building --

on using jsonl which currently CMS is providing examples for both xml and jsonl that allows for better streaming consumption of these large files. Ultimately the tradeoff there is it would significantly increase the size of the file by quite a bit, so something to think about. There have also been suggestions in using like park Kay which is an Apache open format that has this cull far compression in mind as a first class citizen, but some of the downsides would be the complexity in producing and consuming these files and just thinking that's something to keep in mind.

Kind of following that last question, is there a preference to json over xml? Can we have a poll of what everyone is doing to learn more about that? It really depends on what the internal development team's --

how it weighs its pros and cons to producing a file. CMS is providing a schema to support compliance with both of those.

There may be benefits on why one would use Xml over json. You have to go back to your develop team to see what would be the best to choose. Personally CMS is taking the approach that json is going to be a better option over xml. We certainly recognize that current systems are still heavily based within xml, and we wanted to provide a schema because of that fact. And a lot of this data that we're looking at within these files has that parent-child object relationships which make the development somewhat complex.

And json naturally -- and xml naturally lend themselves in the representation of those relationships in a somewhat concise manner.

There's some pretty basic questions such as what is a schema. Those type of questions would be best answered on the transparency and coverage website where we could have some of those basic question-and-answer situations.

Very quickly, a schema is a definition of the format of a filefile.

file.

Can you speak to how frequently you will be providing these pre-version 1.0 releases? CMS envisions -- within the last few weeks engagement within GitHub has definitely stepped on the gas, and CMS envisions that to continue. There are major buckets with most of these --

where most of these discussions fall under. We see that once addressing some of these larger buckets where the majority of these questions are under -- we see the actual production and iteration of these files to actually slow down. So we could see a continued development in the way it's been over the past month for potentially the next month, two or three based on the feedback we're getting on GitHub through discussions and also through these webinars to make sure we're not missing anything that's of concern.

Is there -- kind of getting back to the versioning, is there a pace defined for a version change, for instance, only one version change in the past 12 months. So version 1.0, it will be set in March regardless of how much activity there still is, either a bunch or most of that activity through discussions have been resolved, it will be set. Introductions and changes to the schema that are not backwards compatible, meaning if there's an introduction to the schema that actually breaks what was previously there, that would trigger a new major version, and that would necessarily -- there wouldn't be plans to do that without actually having future rule making in process -- future rule making planned or a PRA planned because, like was mentioned earlier in the call, compliance is really going to be defined on 1.0. So we don't want to have a moving target for developers and consumers of the file.

More of a signal to what to expect coming up, on some of the things that we're working on, it's been noted that professional services have --

will service -- the service code is really directed towards professional services, and that's true. CMS is looking at contextually adding attributes to flag both claims and negotiated rates that will allow the developer and the consumer -- or the developer to indicate whether it's a facility or non-facility place of service to help capture those non-professional services.

I think most of these questions we're going to have to take back and answer. Elissa, we're almost out of time. That's all the time we have for the live Q&A today.

These questions are gathered and saved, and the team, the transparency in coverage team will certainly be responding to them. If you didn't receive a response to your question, we are going to do our best to send out an answer to you directly or to the larger group on GitHub and the transparency and coverage website.

With that said, Elissa, any final words?

>> ELISSA DINES: Thanks, Scott.

I really appreciate everybody's participation today. Really good questions coming in. One question I will address is can we consider Scott the official word of CMS? And you can. We're working on a picture of Scott in a CMS outfit. But for now associate his name with CMS, and when he weighs in on GitHub, you can take that as official CMS direction.

Okay. So with that, an email invitation for the next webinar will be sent out soon once we kind of look at poll responses and some of the feedback from the Q&A box and figure out what the topic will be, you'll be able to register on the GitHub site or the TIC site. Another reminder to provide questions ahead of the call on GitHub to the extent you can so we can prepare some thoughtful answers for the webinar.

Okay. With that, have a great day and a lovely weekend. Take care.