

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE, LLC,
Petitioner,

v.

DIALECT, LLC,
Patent Owner.

IPR2024-00752
Patent 8,849,652 B2

Record of Oral Hearing
Held: August 5, 2025

Before KRISTEN L. DROESCH, HYUN J. JUNG, and
GEORGE R. HOSKINS, *Administrative Patent Judges*.

APPEARANCES:

ON BEHALF OF THE PETITIONER:

HEIDI L. KEEFE, ESQ.
Cooley LLP
3175 Hanover Street
Palo Alto, CA 94304
hkeefe@cooley.com; (650) 843-5001

ON BEHALF OF THE PATENT OWNER:

WAYNE M. HELGE, ESQ.
JAMES T. WILSON, ESQ.
Bunsow De Mory LLP
701 El Camino Real
Redwood City, CA 94063
whelge@bdiplaw.com; (571) 208-0186
jwilson@bdiplaw.com; (571) 222-1022

The above-entitled matter came on for hearing on Tuesday, August 5, 2025, commencing at 1:00 p.m., via video teleconference.

P R O C E E D I N G S

- - - - -

JUDGE HOSKINS: Thank you. So, we're on the record now.

This is a hearing for oral argument in IPR2024-00752, concerning challenged U.S. Patent Number 8,849,652. I am Administrative Patent Judge George Hoskins. And appearing on the panel with me, in the hearing room, so you see them in one video feed, are my colleagues, Kristen Droesch and Hyun Jung. And so, with that, could I get an appearance from counsel for Petitioner in this hearing, please?

MS. KEEFE: Yes. Good afternoon, Your Honor. This is Heidi Keefe from Cooley, representing Petitioner Google, LLC.

JUDGE HOSKINS: Thank you. And welcome to the Board today again, Ms. Keefe.

MS. KEEFE: Thank you. Nice to see you again.

JUDGE HOSKINS: And counsel for Patent Owner, please.

MR. HELGE: Hi. Good afternoon, Judge Hoskins. Wayne Helge from Bunsow De Mory Law Firm for the Patent Owner, Dialect, LLC. Judge, in the conference room with me here also is my backup counsel, James Wilson. Although, as with Friday, I don't expect him to be appearing on video or speaking to the panel today.

JUDGE HOSKINS: Very good. And welcome back Mr. Helge.

MR. HELGE: Thank you, Judge.

JUDGE HOSKINS: I'm going to try to keep the introductory part here brief since we just went through this on Friday. Just, a few of the differences today versus Friday, today is 45 minutes per side so it's a little bit longer. And so, with that, I will pause after the two case in chief

1 presentations are done to see if anybody needs a little break. So, Petitioner
2 will have its case. Patent Owner will have its case, then I'll just see if
3 anybody needs a little bit of a break. If you do, that's great. If you don't,
4 that's great. That's just kind of your opportunity at that point.

5 And then secondly, you can see, I think, above my colleagues in
6 the hearing room, a clock that currently says 45. Hopefully you can see that.
7 So, that's going to be your official timekeeper today. I'm going to ask for a
8 reservation for rebuttal when you begin your case in chief, and so the clock
9 will then be modified to reflect that. When you reach that reservation, you
10 don't need to stop your case in chief unless you want to. But the
11 consequence, of course, to continuing is, you're eating into your rebuttal
12 time.

13 And then, finally, just a little reminder, that when we're done with
14 the argument today, please stick around for a few moments to make sure the
15 court reporter gets whatever information the court reporter needs.

16 So, with those brief introductions, are there any questions about the
17 procedure here today?

18 MS. KEEFE: No, Your Honor.

19 MR. HELGE: No, Your Honor.

20 JUDGE HOSKINS: Okay, thank you. So then, Ms. Keefe, how
21 much time would you like to reserve for your rebuttal?

22 MS. KEEFE: I don't think I'll need this much time for the opening,
23 but I'll reserve 15 minutes.

24 JUDGE HOSKINS: Fifteen minutes. Okay, thank you, so noted.
25 And so, you can see the clock now says 30 for your case in chief. And so,
26 when you're ready, I welcome you to begin. Thank you.

1 MS. KEEFE: Thank you, Your Honors. There are only two issues
2 that remain for us here today in this petition: whether Coffman teaches
3 synchronization of context stacks, as recited in the claims and supported by
4 the specification, and whether Coffman teaches a system that identifies a
5 domain agent, or application, associated with a context entry and generates a
6 response using that agent or application. The answer to both is yes.

7 Both the patent and Coffman are trying to enhance the processing
8 of natural language utterances, or questions or commands, by also using
9 context to try to get the right information to the user as quickly as possible.
10 For example, in the '652 Patent, Exhibit 1001, an example is given where the
11 user is asking, show me all of the photos of Greece. This is in Exhibit 1001,
12 column 21, line 27.

13 In the patent, the system there parses the words and knows to limit
14 itself to photo applications and tries to find photo objects within that photo
15 application. It also parses the word Greece to determine that that's a
16 location, and looks for a GPS application in order to find out where Greece
17 is so they could return the results to the user of photos from Greece.

18 Coffman is accomplishing essentially the same thing, and uses a
19 similar request to check a spreadsheet while a user is simultaneously
20 checking email. This is in our Slide 71, which shows the quote from page
21 56, lines 11 through 15. Patent Owner's expert, on Slide 72, essentially
22 agrees that this is what Coffman is doing, and in fact, he says that once the
23 speech recognition of the CVM assesses and determines that it's a particular
24 spreadsheet and particular cell, then the application would access the
25 spreadsheet and the cell again, using the CVM and its resources would
26 create a verbal response.

1 So, if we switch now to Slide 7, we know that Coffman, like the
2 '652, is trying to allow user to, and I'll quote, "seamlessly control multiple
3 Windows applications, for example, running in parallel." And this is at
4 Coffman page 4, lines 13 through 16, again Exhibit 1003.

5 So, how does Coffman work? I want to just take a step back and
6 show how Coffman works. If we turn first to Slide 8, we see that Coffman
7 discloses a conversational user interface, which interacts with
8 conversationally-aware applications, like the spreadsheet and email program
9 that were discussed a few seconds ago. In turn, the conversationally-aware
10 application interacts with a conversational virtual machine.

11 Turning to Slide 10, the conversational virtual machine, as shown
12 in Figure 1 on the left, is comprised, as shown in Figure 4 on the right,
13 essentially a context stack with a dialog controller. Coffman tells us the core
14 of the CVM is the context stack, which operates and is managed under the
15 direction and control of the dialog controller.

16 Turning now to Slide 11, this is the text that accompanies that
17 figure. We're told that Coffman tells us, at page 29, lines 8 through 32, that
18 its context stack includes all of the queries, commands, and activated tasks
19 or applications, just like the '652, Exhibit 1001 at column 20, lines 10
20 through 12 and column 36, line 13, which indicate that its context stack
21 includes the queries and activated users that are being done in that patent as
22 well.

23 Turning to Slide 14, Coffman tells us that each time a new query or
24 utterance comes in, the system will match down the context app. It will
25 search down the context app, looking for matches to try to find the right
26 application to activate. That value, or context, is then pushed to the top of

1 the stack so that it can be top of mind for the next search. And this is in
2 Coffman at page 42, lines 1 through 7.

3 That all works really well when you have --

4 JUDGE HOSKINS: Ms. Keefe?

5 MS. KEEFE: Yes.

6 JUDGE HOSKINS: One of the things you said is, that the context
7 stack is used to identify an application --

8 MS. KEEFE: Yes.

9 JUDGE HOSKINS: -- to invoke, I think?

10 MS. KEEFE: Yes.

11 JUDGE HOSKINS: Can you show me where that is in Coffman?

12 MS. KEEFE: So, for example, right in -- sorry, I just want to give
13 you the exact right one here. Hang on one second. So, for example, on the
14 slide that we actually had up in front of us, right in front, in Slide 14,
15 Coffman specifically talks about searching down the context stack for the
16 active command that's completed or a new command that can be activated,
17 and so that's one form of showing that.

18 The other -- if I turn to Slide 18, this is in Coffman, page 25, lines
19 10 through 14, context information may be transmitted by indicating passing
20 or pointing to the context stack history of the device or application that the
21 controller can access and add to its context stack. So, that's saying that the
22 application is something that can actually be in -- activated and added to the
23 context stack. Does that help, Your Honor?

24 JUDGE HOSKINS: I suppose. I'm asking, of course, because of
25 Claim 8.

26 MS. KEEFE: Yes.

1 JUDGE HOSKINS: I don't know that this Slide 18 says that the
2 context stack is used to invoke one of those applications, the
3 conversationally-aware applications, that you colored green on your Slide 9.
4 That's really the question I'm trying to get at.

5 MS. KEEFE: Okay. Well, I think one of the other things that
6 might be very helpful is, if we look, for example, to Slide 47. Slide 47 is
7 actually Claim 11 from Coffman itself. And Slide 47 specifically points out
8 that the dialog manager, which is a part of the CVM, which is interacting
9 with the context stack, tells us that it's being used to select an application
10 based on those input queries. And so, those applications, which we know
11 from what I just said, are added to the context stack, are what is being
12 actively selected. And so, I think, actually Claim 11, which is in Coffman at
13 page 65, lines 12 through 13, is very helpful there. And similarly, in
14 Coffman, at page 31, lines 24 through 29, we see that the dialog controller,
15 which we know is managing the context stack, manages and determines the
16 active context and application, and so those are in the stack.

17 I think, finally, just to give one more example, if we look to, for
18 example, Slide 52, we also see that Slide 52 points out Coffman at page 42,
19 lines 27 through 29, which indicates that the context stack is updated
20 pending on voice, keyboard, mouse, or any other input or command and on
21 the application output. And so, that application is updating the context stack
22 as well. I hope those are helpful, Your Honor.

23 JUDGE HOSKINS: Thank you.

24 MS. KEEFE: Thank you. And again, I'd like to just point out to
25 remind Your Honor, that the whole point of the patent is to allow a user to
26 seamlessly interact between applications. And so, the whole point is to

1 know which application you're going to interact with, and those applications
2 are stored in the stack, so as you manage down a stack, you're trying to find
3 the right application to invoke.

4 The example that Patent Owner gives, when it argues against
5 Claim 8, is a secondary embodiment wherein an application can have its
6 own CVM, but not the embodiment that we mapped, wherein the devices
7 each have their own CVM, which includes all of the application lists. And
8 that's how, when a user is running their email program, they can say, and
9 give me the spreadsheet information, and then that is what takes it back into
10 the system to try and find the spreadsheet and find that information as well.

11 So, if we go back and look -- in fact, that's actually a helpful place
12 for us to switch to Slide 16, which is actually Figure 10 of the patent. In
13 slide -- sorry, in Figure 10, I actually used Your Honor's annotated version
14 as opposed to the one we annotated. We annotated one that just had the
15 yellow boxes highlighted, but Your Honors did a better job of making it a
16 little cleaner and clearer as to all the different things that were involved.

17 And here what we see is, that the point of Coffman is to have an
18 active stack up on time where the master device communicates with any
19 number of other devices. Now, Coffman is clear. Those could each be
20 individual applications, which have their own CVM, or they can be devices,
21 which have their own CVM. The embodiment that we mapped was the
22 devices, each having their own CVM, and those devices would have
23 applications on them, which would be a part of the context stack that would
24 be searched to figure out which application could be used in order to resolve
25 a query.

26 If we then look, for example, to Slide 18, the master/slave, can

1 then update the context stacks of each device with the context stack of the
2 other. The purpose here, and we know this from Coffman at page 23, lines 7
3 through 9 -- I apologize, I don't have a slide for that. But Exhibit 1003 at
4 page 25, Coffman says that the distributed conversation of protocols allow
5 the exchange of information to coordinate the conversation involving
6 multiple devices, or applications, including master/slave conversational
7 network, et cetera. And so, we know here, that the whole purpose of
8 updating these stacks so that they have matching information to each other,
9 is to coordinate these conversations.

10 Slide 18 shows us that the context information may be transmitted
11 by indicating passing or pointing to the context stack of the device or
12 application that the controller can access and add to a stack. And again, I
13 think that helps show that the applications are in the stacks.

14 Turning to Slide 19, Coffman is very clear that the information that
15 can be exchanged between the master and the slave, using the dialog
16 moderator, is and includes -- this is on Slide 19, additions to contexts,
17 context updates, notification of contexts changes, or contexts and state
18 updates due to action, i.e., I invoked that application and therefore pushed it
19 to the top like we saw just a second ago.

20 Despite this clear teaching of updating context stacks based on
21 other context stacks, Patent Owner somehow claims this isn't enough. But
22 let's look at the language of Claim 1 itself. I'm going to Slide 32. At Slide
23 32, the limitation is, that the system must synchronize the first context stack
24 with a second context stack, associated with a second device, here critically,
25 such that the context information of the first context stack is updated based
26 on related context information of the second context stack.

1 So, the limitation is telling us what it means by synchronizing.
2 And it means, that the context information of the first context stack is
3 updated based on related context information of the second context stack. It
4 doesn't say, based on all. It doesn't say, with a perfect synchronization.
5 And, in fact, that's not required by the specification either.

6 The specification, in column 17, clearly points out that the
7 synchronization -- this is Slide 34, the synchronization process -- and I
8 apologize, the one line I liked the best, I didn't highlight until I was getting
9 ready for the hearing today. So, my favorite line is actually the second
10 sentence on the left, the synchronization process attempts to determine
11 which version of a data element, or an agent, are the newest or most up-to-
12 date, and propagates that element. It doesn't say, propagates all the
13 elements. It doesn't say, go back and match things, or make sure they're all
14 in perfect synchronization. It says, figure out what's new and add that.
15 Propagate that element. Bring that over. And, in fact, that same section
16 contemplates that synchronization may be an iterative process.

17 And this was actually also cited by Your Honors in the institution
18 decision, pointing out that that's all the specification requires. That it can be
19 as simple as adding information from one context to another, to make sure
20 that at least that top information matches. So, in fact, the specification not
21 only does not require perfect synchronization, it actually absolutely includes
22 embodiments that do not replace the entire stack. And, in fact, that is
23 alternatively mentioned in that same area of the specification.

24 Moving on to Slide 35, this is the *Dayco* case that we cited in our
25 briefing. In the *Dayco* case, almost exactly like our case here, was a case
26 where the word received was used. *Dayco* involved two hoses that had to fit

1 together, and whether or not the threads worked together, and how far they
2 had to go into each other. And in *Dayco*, the claim construction down below
3 said that it had to be completely received. But when the Federal Circuit
4 looked at it, they looked at the specification and realized that there was no
5 embodiment for completely receiving, and instead, there was simply the
6 possibility of entering in some fashion. So, the part of the hose that was
7 received was not completely received. And so, like in this case, the
8 specification does not support adding a word into the claim limitation that
9 doesn't exist there.

10 One last thing though that we'd like to say is, and this is in
11 response solely to the arguments made by Patent Owner that we did not have
12 a chance to reply to until our reply papers. Even if perfect synchronization
13 were required, one of ordinary skill in the art would understand that
14 Coffman does, in fact, teach perfect synchronization, especially in instances
15 where the stacks are very small. For example, in the first two or three pieces
16 of information, and the master may only have two, the master sends those
17 two down to the slave. The slave gets one thing added and tells him after,
18 hey, go and add that one more thing, you will have perfect synchronization
19 of those elements. And that was expressed by our expert at Exhibit 1017,
20 paragraph 11, and is explained further in our reply at pages 7 through 8.

21 JUDGE HOSKINS: Ms. Keefe?

22 MS. KEEFE: Yes.

23 JUDGE HOSKINS: In that regard, can you help me out with
24 respect to the order of the entries in the context stack? One of the things that
25 Patent Owner spends a fair amount of effort on is an attribute of a context
26 stack is the order of the entries. And so, if we're going to synchronize to

1 context stack, one of those aspects of synchronization is the order of the
2 entries.

3 MS. KEEFE: So, I have two answers to that, Your Honor. The
4 first answer is actually, Patent Owner itself did not -- Patent Owner itself, I
5 should say this, in Exhibit 3001 at page 21, Patent Owner is trying very
6 oddly to have it both ways here. So, at Exhibit 3001, page 2, Patent Owner
7 argued, persuasively, to the District Court, the context stack, as used in the
8 patent, does not follow a specific order. And, in fact, it does not have to be
9 anything more than a collection of contexts. And so, there was not an order
10 imposed. And they emphatically argued that to the District Court,
11 persuasively so.

12 The District Court accepted that definition. And yet, here today,
13 they're trying to say, that it mandates an order, and the order has to be
14 matched. That's absolutely not true. And again, as indicated by column 17
15 of the patent, you just have to make sure that the newest one matches up.

16 I think, some order might be applied by stack, but it doesn't have to
17 be FIFO all the way down, or LIFO, or anything like that. Instead, what we
18 see in Coffman is that the last thing used will be pushed up to the top of the
19 stack, and the patent talks about matching the two top elements, or the most
20 recent new things, and making sure that those are used for updating and
21 synchronizing. I hope that answers Your Honor's question.

22 JUDGE HOSKINS: So, two follow-ups. One is just simply -- you
23 referred to Exhibit 3001, and I think you were referring to Exhibit 3002
24 instead. That's just kind of a --

25 MS. KEEFE: I apologize, Your Honor.

26 JUDGE HOSKINS: No worries. But the second one more

1 substantively is, let's suppose we adopt what Patent Owner is urging us, as I
2 understand it in this proceeding. And we'll see when your colleague gets up,
3 what he has to say on this. But let's suppose that I understand the argument
4 right, and part of the synchronization has to be order of the entries in the
5 context stack. So, let's just suppose, from a claim construction perspective,
6 that's where we end up. I don't think that you've addressed that specific
7 possibility of order in terms of how the prior art combination you've given
8 us can lead to the obviousness of synchronizing order. Am I correct on that?

9 MS. KEEFE: No, I think we actually have. For example, Your
10 Honor, we have, showing, for example, at Slide 37 and 38, we did absolutely
11 address that even if some form of perfect synchronization, including order
12 were mandated, that that would happen when a person of ordinary skill in
13 the art understands that, for example, the master -- take an example, as we
14 discussed in our reply at pages 7 through 8, if you have a -- relatively early
15 in the stack, there's only a few elements that are in order, and then the slave,
16 for example, says, hey, I just topped application number two up to the top of
17 my stack, as Cofflet (sic) talks about. Cofflet says, "When you activate an
18 application, it is pushed to the top of the stack."

19 Then, that is sent back up to the master, the new one is at the top,
20 therefore the new one needs to be added. That will be added to the top, that
21 will match. And so there, the order will match as will everything else. And
22 that's very explicitly talked about in our papers at pages 7 through 8. And I
23 think I misspoke, I meant Coffman, because Coffman is the one that's
24 popping it up to the stop of the stack. And so, they would actually match in
25 that case.

26 JUDGE HOSKINS: Thank you.

1 MS. KEEFE: Thank you, Your Honor. So, I think I could -- if
2 Your Honors don't have any other questions about synchronization, I'll
3 switch to the last elements of identifying and generating, if that's okay.

4 So, for identifying the domain agent, what I'd like to point out first
5 is that the domain agent -- this is on Slide 41 of our materials, the domain
6 agent is actually -- I think this is undisputed, the conversationally-aware
7 application. So, the domain agent is the executable software that is being
8 found in the -- it's that executable software that's going to be executed in
9 order to carry out the clarity of the request. Coffman has a conversationally-
10 aware application, and we see that in the petition at page 54, where we're
11 citing Coffman, page 13, lines 26 through 29, calling out the
12 conversationally-aware application as the code implementing the
13 applications.

14 If we then move forward to -- and again, let's never forget, as Slide
15 7 tells us, that the whole point of Coffman is to seamlessly control multiple
16 Windows applications, and the way you do that, is you have a system that
17 allows you to choose, and select, and activate those applications based on
18 queries.

19 So, at Slide 42, how does Coffman choose the right application?
20 Slide 42 shows us in Coffman, at page 41 and page 37, that the system
21 marches its way down -- the output of these subsystems is run down the
22 context stack to extract the active query and complete it. So, figure out what
23 you need to do. As soon as a context is found that fits the utterance at the
24 level of the NLU symbolic language, the context becomes active, in other
25 words, the application is activated, and the corresponding process becomes
26 active.

1 Now, I can use my spreadsheet. Now, I can use my email
2 program, whichever it was. The first task to have its dialog completed by
3 producing a non-ambiguous query is executed. So, it is selected and
4 executed.

5 I would again turn Your Honors, emphatically, to Slide 47, which
6 is Claim 11. This was so important to the invention of Coffman that
7 Coffman claimed it by saying that the dialog manager is for managing dialog
8 across applications and for selecting an active application based on input
9 queries. So, we think that's very clear.

10 Patent Owner seems to be conflating or confusing -- and this is at
11 Slide 45, a conversational user interface with the conversationally-aware
12 applications. Applications are different from the interface, although they
13 seem to be pointing constantly to this is just an interface. That's not true, the
14 applications are separate from that.

15 And in fact, if we look to Slide 46, again, reminding us that the
16 whole point is to control multiple Windows applications using that dialog
17 manager, the claim telling us that we're going to select which application we
18 want.

19 We then go to Slide 50, which reiterates that it's the applications
20 that are being managed, and Coffman equates applications with tasks,
21 processes, and discourses.

22 And we see on Slide 51, we've actually shown through a graphic
23 here that, while originally, when describing the CVM to have the ability to
24 dialog with context and mixed-initiatives across tasks, processes, and
25 discourses. Later, in the exact same disclosure, they talk about that exact
26 same interaction happening with multiple applications, so they're using those

1 words interchangeably.

2 One of the last things I'd like to point out is, again, the language
3 that Patent Owner seems to like the most, and this is on Slide 57, is that
4 Coffman -- typically different devices, CVMs, or different applications will
5 have their own context stack, they seem to take that to mean that every
6 application has to have its own context stack. That's not true at all.
7 Coffman has alternative embodiments, and we can see that in Slide 16,
8 which shows Figure 10, where each of the boxes down below that are being
9 managed, can either be an application or a device. We mapped the
10 embodiment, where the context stacks are on the device, and therefore those
11 context stacks have the applications, not are the applications.

12 Finally, Your Honors, so that shows why the Claim 11, especially,
13 points out that Coffman specifically talks about identifying the application
14 that's going to be used. The limitation then requires that the system generate
15 a response using that agent. And the best evidence we have for that is Patent
16 Owner's own expert, at Slide 72, admitting that in the example of the
17 spreadsheet, if the spreadsheet is invoked through a query, the spreadsheet
18 application is found, and then the system will use that application to generate
19 a response, which is passed back through the user interface to create speech
20 so that the response can be made. Unless Your Honors have any other
21 questions, I think I'm right at time.

22 JUDGE HOSKINS: Thank you. I do not, and I'm not hearing any
23 from my colleagues, so thank you. Mr. Helge, would you like to reserve any
24 time for your rebuttal period?

25 MR. HELGE: Yes, Your Honor. Please, if I could, just reserve
26 five minutes. And, Your Honor, just for housekeeping, we do have a lot of

1 ventilation noise in this space. I want to make sure you can hear me and
2 you're not hearing a lot of background hum. Is it okay here?

3 JUDGE HOSKINS: I could hear you perfectly fine, and I do not
4 hear any background hum, so I would not lose any sleep over that.

5 MR. HELGE: Okay, thank you, Judge. Whenever you're ready
6 I'm happy to begin.

7 JUDGE HOSKINS: So, please begin when you're ready. Thank
8 you.

9 MR. HELGE: Thank you, Your Honor. May it please the Board,
10 Your Honors, because we sort of book-ended Petitioner's presentation with
11 the Claim 8, another dependent claim discussion, I think it makes sense for
12 me to begin with those claims as well.

13 What I'd like to do, Your Honors, is share -- I know that -- well, if
14 Your Honor would like me to share my screen, I'm happy to do it. I'm going
15 to be going off of my slides, if I may.

16 JUDGE HOSKINS: Whichever makes you most comfortable. We
17 have your slides in front of us, but if you would like to walk us through
18 them, that's fine as well.

19 MR. HELGE: Okay, Your Honor. That's what I'll do then. I'll
20 just walk you through them. So, what I'll be doing is, beginning on Slide 21.
21 And I think Petitioner's Counsel made a very big point about Claim 11, and
22 it's important to note that we did address Claim 11 in our sur-reply. And
23 Claim 11 does not support the position in the read of Coffman that Petitioner
24 is espousing.

25 So, beginning at Slide 21 of our presentation, we have some
26 discussion of our dependent Claims 8, 18, and 24, and this also applies to a

1 few claims that depend as well from these claims. We have the claim
2 language, identify one or more domain agents associated with the one or
3 more context entries, wherein the one or more domain agents are configured
4 to process the command or request.

5 And one thing I want to note, Your Honors is, Petitioner said,
6 when she pointed to her Slide 41, that she doesn't think that there's a dispute
7 whether the conversationally aware applications of Coffman disclose or
8 suggest the domain agents of this claim, and that is absolutely in dispute.
9 But we disagree, the Board actually went against Petitioner's interpretation
10 in five other IPRs, one of which we included as an exhibit, and it was the
11 IPR-202400746 case. But there were five IPRs where Petitioner made this
12 argument that Coffman's conversationally-aware applications, taught, or
13 disclosed, or suggested domain agents as appearing in these claims, and
14 that's not right. The Board correctly found that that was not an accurate
15 characterization.

16 Moving to my Slide 22, importantly, Claim 11, the Petitioner
17 keeps focusing on the dialog manager aspect of Claim 11. And what I'd like
18 to do, I'll point into Coffman's disclosure itself, which explains what the
19 dialog manager does which, in fact is, routing queries, or commands, or
20 requests from a user that have come in, into the CVM, routing those, and
21 understanding which application they're coming from. And that's why, in
22 Claim 11, the dialog manager has to understand and be able to identify
23 which application is associated with a command request.

24 But what Petitioner never addresses -- they didn't address it in their
25 reply, they certainly didn't address it in their petition in the brief discussion
26 of Claim 8 that they included in the petition, is that Claim 11, as we

1 highlighted on the second highlight on Slide 22, Claim 11 recites a
2 conversational task dispatcher for coordinating and dispatching
3 conversational tasks. It is not the dialog manager of Coffman that figures
4 out how to process a command or a request that comes from a
5 conversationally-aware application through the user interface to the CVM,
6 it's the conversational task dispatcher.

7 So, the indication or the recitation in Coffman's Claim 11, that the
8 dialog manager has to identify an application or, of course, select an
9 application, it's not about figuring out which application has to carry out a
10 command, it's about figuring out which application it came from, and being
11 able to put it in the context stack correctly. So, as we say Claim 11 recites a
12 conversational task dispatcher.

13 Now, if you were to go to --

14 JUDGE HOSKINS: Mr. Helge?

15 MR. HELGE: Yes, Your Honor.

16 JUDGE HOSKINS: Can you help me find the dispatcher that
17 you're talking about in the figures?

18 MR. HELGE: Absolutely. So, in Claim 11 -- I can put it on my
19 screen if I need to. So, Your Honor, in Claim 11, the first element, it's a
20 system of Claim 1. And this is -- again, this Claim 11 is Coffman's Claim
21 11, Your Honor, not our patent. So, the first element is the dialog manager,
22 which is what Petitioners are leaning into and saying, well, look, the dialog
23 manager has to select an active dialog, context and application, but that's not
24 selecting a domain agent, because a conversationally-aware application is
25 not a domain agent. But then if you go to the next element, you have a
26 resource manager. The next element is a conversational task dispatcher for

1 coordinating and dispatching conversational tasks.

2 So that's the aspect of the conversational kernel, or the CVM, that
3 figures out, how are we carrying out these requests that come in from a user
4 via conversation, via vocal commands?

5 So, here's the important part, Your Honor. If you were to look, and
6 I can read it to you, but if you have it available, Coffman page 31 -- now,
7 this is confusing because it's actually 33 of the exhibit, but page 31 at the
8 bottom center of the Coffman Exhibit 1003, beginning on line 15 says that
9 the task dispatcher, which is what we were just talking about from Claim 11,
10 feeds input from the conventional and conversational subsystem services,
11 412, 406 to the context stack 405, then there's a parenthesis (via the dialog
12 controller 404, which selects the active context). And here you go. And
13 feeds the output of the different tasks to the different subsystems and
14 prioritizes them.

15 What that's telling you is, that you're going to the subsystem in
16 Coffman to process these tasks. You're not receiving a task at the CVM,
17 figuring out an application to carry it out, spitting it back out to the
18 application and then getting a response. Coffman talks about carrying these
19 out at the subsystem.

20 Now, this was consistent with Petitioner's theory. Because if you
21 look at page 57 of their petition, which deals with Claim 8, they point to
22 Figure 6 of Coffman. Now, Coffman's got a lot of embodiments. There's a
23 lot of embodiments that, frankly, I think nobody has really quite figured out
24 how they all necessarily interoperate. But if you look at Coffman on page
25 57 -- I'm sorry, Petition page 57, where they annotate Figure 6 of Coffman,
26 there's a back-end application, 608, that is available for carrying out and

1 figuring out responses to these commands. This back-end application has
2 always been Patent Owner's viewpoint that Coffman's CVM sends
3 commands, queries, requests from the stack, down to back-end -- I'm going
4 to call it a general back-end, and here it's reflected by the back-end
5 application 608, to carry these things out. And then it spits out a vocal or
6 verbal response to the user, and we see that in Figure 6.

7 So, this is what they've charted for Claim 8, and this is consistent
8 with Claim 11. The task dispatcher is figuring out how to send a request,
9 query, command to the back-end to process. This isn't a situation where
10 Coffman's CVM is receiving a verbal command, request, query from a user,
11 figuring out another conversationally-aware application to carry that out,
12 coming back to the CVM, and then kicking it back out to the user. That's
13 just not how Coffman operates. Figure 6, what they charted, is consistent
14 with our read of Figure 11, you have to read past the dialog manager, which
15 is where their read began and ended, and it's not an accurate read of
16 Coffman.

17 If I were to go -- I sort of spoiled my own Slide 23. And Slide 23
18 is actually where we made the point in our sur-reply, that the task dispatcher
19 feeds the output of the different tasks to the different subsystems and
20 prioritizes them. The task dispatcher is carrying this out, not the dialog
21 manager that they're pointing to in Claim 11. So, I agree with Petitioner,
22 Claim 11 is very important for answering this question, but it's not the way
23 the Petitioner reads it. It's the way that -- we've explained it, when you read
24 the entire claim, the task dispatcher answers that question.

25 Your Honor, just for closing this topic out, Slide 24 simply says,
26 and identifies all the claims that this issue addresses. And I think if Your

1 Honors agree with us, dependent Claims 8, 18, 24, and the respective
2 dependent Claims 9, 11, and 25 through 27 survive this IPR regardless of
3 what happens in the synchronization aspect. And if Your Honors have no
4 other questions on this point, I will move to the beginning of my slide
5 presentation and discuss the synchronization aspects.

6 JUDGE HOSKINS: I do have one question for you before we
7 move to that.

8 MR. HELGE: Yes, Your Honor.

9 JUDGE HOSKINS: So, trying to find a good slide for this. So,
10 Figure 1 of Coffman, can you pull that up --

11 MR. HELGE: Yes, Your Honor.

12 JUDGE HOSKINS: If you have that handy.

13 MR. HELGE: Mm-hmm.

14 JUDGE HOSKINS: Thank you. So, I'm looking at -- we've got
15 two kinds of applications here. We've got, quote, "Con conversationally-aware
16 applications 11", which communicate with the CVM through some APIs.

17 MR. HELGE: Mm-hmm.

18 JUDGE HOSKINS: And then we also have conventional
19 applications 12. I think you spent a fair amount of time on kind of these two
20 different applications in your Patent Owner Response. And I never quite
21 understood how that fits into either your arguments for Claim 8, which it
22 seems to me is mostly the more likely of the two, but how it fits into your
23 arguments for Claim 8, or when you turn to it in Claim 1. So, if you could
24 just help me out with that, I would appreciate it.

25 MR. HELGE: Certainly, Your Honor. So, I think there's one thing
26 that's critical about Coffman's claim -- or Figure 1, that is very interesting in

1 terms of what is Coffman really trying to achieve? From the
2 conversationally-aware application API, 13, and the conventional apps 12,
3 from there down, the conversational virtual machine, it handles everything
4 the same. In other words, the only thing that's special about
5 conversationally-aware applications is they can communicate via
6 conversationally-aware application APIs to communicate with the virtual
7 machine, the CVM. You also have conventional apps that can communicate
8 with the conversational virtual machine.

9 And so, I think what's important about this is, if you were to take
10 the conversationally-aware applications and the conversationally-aware
11 application APIs and combine them, and in fact you have an application.
12 That's all you have is an application. It's something that can speak to the
13 machine, can go down, send commands, things can be processed. And, in
14 effect, from the conversational virtual machine's point of view, whether
15 you're getting a conventional application's query via keyboard or mouse
16 click, or whether you're getting a person saying something, from the
17 conversational virtual machine, there really isn't much of a difference.
18 Here's a command. Here's a query. Here's something we have to process
19 and we have to spit back and answer.

20 I'm not sure if I'm answering Your Honor's question. But I think
21 the important thing here -- and we do have, obviously, context 7 -- block 17
22 has some additional aspects that can speak with the conversationally-aware
23 APIs. But there's a lot of focus in the petition about, for example, this
24 discussion of, well, you can access a spreadsheet while you're on your email.
25 Now, that doesn't talk anything about what's going on in the stack, right?

26 When you get into these latter portions of Coffman, like, page 56,

1 talking about -- or you're driving to the store, and you want to figure out
2 your shopping list, or modify your shopping list, Coffman never really
3 explains to you what's going with the stack in those situations, right? All
4 you're doing is, is you're talking about, can you interact with the
5 conventional app? Can you interact with the conversationally-aware
6 application? And the answer is, because you have a UI at the top, of course
7 you can access both. But it doesn't explain what's going on in the stack, how
8 the system is using the stack to -- for example, process requests, commands,
9 queries, things like that. The conventional apps and the conversational apps
10 are simply part of the software suite that's available in the system.

11 JUDGE HOSKINS: Thank you.

12 MR. HELGE: I think it's a long-winded answer, Your Honor. But
13 I think the answer is, Coffman has one additional layer so that a user can talk
14 to an application. And I think that's -- at least in terms of the upper end,
15 that's it. The lower end is really where all the activity is happening in
16 Coffman.

17 So, Your Honor, if I may, I will go back to my Slide 2 -- actually,
18 yes, my Slide 2, which just has Claim 1. I know we've already talked about
19 this. I just would like to reiterate a brief point here and raise a phrase that I'd
20 like to think about as we go through this discussion. The claim language at
21 issue was synchronize the first context stack with a second context stack
22 associated with a second device, such that the context information of the first
23 context stack is updated based on related context information of the second
24 context stack.

25 Now, the question that I'd like to pose today, and I think it's
26 important, because we need to understand, first of all, what does this phrase

1 encompass? What does this claim language encompass? And the phrase I'd
2 like to throw out is who's driving the bus? When Counsel pointed to her one
3 slide with that claim language -- I think it was Slide 32, highlighted "the
4 synchronizing, in effect, disappeared." What was going on with
5 synchronizing became subservient to the updating. In Petitioner's current
6 view, the updating is driving the bus. If you update, there must be
7 synchronizing.

8 Our view is different, and I think Judge Hoskins, I think you
9 recognized this with some of your questions to counsel. You have to
10 synchronize, and that synchronizing results in updating, and that's why that
11 language such as that appears.

12 Now, what's interesting about this, and I'm going to skip to Slide 3,
13 Your Honor, which just recites the claim language of Claim 14 and Claim
14 20. For the purposes of this IPR, Petitioner has not addressed any
15 differences in these with respect to these claims. These will be Claim 1.

16 JUDGE HOSKINS: Hold on Mr. Helge, I just want to make sure
17 we're using the same slide deck. Because the Slide 3 that I'm looking at for
18 your slides has Claims 3 and 8 as opposed to Claims 14 and 20, so.

19 MR. HELGE: Your Honor, I'm on Exhibit 2016. Is that what you
20 have as well? No.

21 MS. KEEFE: Mr. Helge, just so you know, my version of your
22 deck at Slide 13 also is 14 and 20.

23 JUDGE HOSKINS: Hmm. So, let me make sure I got the right
24 thing. So, Exhibit 2016, right, 2016?

25 MR. HELGE: Yes, Your Honor.

26 JUDGE HOSKINS: Okay. I was wrong. I apologize. Thank you

1 very much.

2 MR. HELGE: No problem, thank you. And thank you, Heidi. So,
3 Your Honor, I'm skipping to Slide 3. We have Claims 14 and 20, just some
4 language there from those claims. When I get down to Slide 4, this is just a
5 very brief recitation of what the Petitioner said about Coffman with regard to
6 this feature, the synchronizing feature. But I think the important thing is
7 Slide 5, and I'd like to focus on this for just a moment.

8 So, this was the conclusion of Petitioner's discussion of
9 synchronizing, where they were moving from synchronizing to updating.
10 And they say here, as I've quoted in Slide 5 from the petition, "A POSITA
11 would have further understood and found it obvious that, as a result of
12 synchronizing device context stacks in Coffman's system," -- then they quote
13 the rest of Claim 1.

14 But I think the important thing here is, Petitioners realize that
15 synchronizing was driving the bus. You had to have synchronization that
16 resulted in updating. And so, we do need -- I don't think we should focus
17 this discussion on updating. The question really is, what is synchronizing?

18 By the way, if we go to Slide 6, this is the reply from Petitioners.
19 And at this point, they have sort of discarded the requirement of
20 synchronizing entirely and focused purely on updating. And this is more
21 consistent with Counsel's argument today, which is, you need to have
22 updating, and that is, effectively, what's driving the bus in their argument
23 here. So, as I have this little note here, their position now is that
24 synchronizing is not needed, only updating.

25 So the reason that I think we need to spend a little time, or that I've
26 decided to spend a little time, Your Honors, talking about synchronization

1 -- and I've moved on to Slide 8 here, it's something that the petition said
2 about claim construction. We all know that 37 CFR 42.104(b), I think it's 3,
3 requires Petitioners to take the position on claim construction.

4 Here, what Petitioner said was that claim construction is
5 unnecessary because the prior art renders the challenged claims obvious
6 under any reasonable construction. But we know based on, at least Mr.
7 Schmandt's deposition testimony, that this is not right. Because what we
8 have on Slide 9 is some of Mr. Schmandt's deposition testimony regarding
9 the meaning of the term synchronized.

10 So, we had asked him a question, on Slide 9, the top portion is the
11 question. Basically, we were asking him about stacks that had identical
12 content but in different orders. And we said AB -- if you have one stack
13 with ABCD and another stack with DCAB, would they be synchronized?
14 And his answer was, it depended on how a POSITA would understand
15 synchronized. And he said, "Which is nuanced by what the application is
16 and what the context is."

17 Now, this is important, because as we saw, Petitioners didn't take a
18 position on claim construction. They didn't tell you that synchronization
19 could be less than perfect. They said, under any interpretation you could
20 come up with, we win. And that's not accurate. Because Mr. Schmandt, as
21 we go to Slide 10, he said that, "Synchronization may or may not require"
22 -- and he stumbled a little bit here, he said, "may or may not be required to
23 be absolutely perfect."

24 Now, how would we know, at this point, whether Coffman teaches
25 synchronization if we don't even know what synchronization means? Mr.
26 Schmandt says, "There are many factors," as we point out in our Slide 11, he

1 spent four pages of deposition transcript testimony discussing factors that
2 would be relevant to how a POSITA would understand the term
3 synchronizing.

4 Now, this didn't show up in the petition. This didn't show up in the
5 opening declaration for Mr. Schmandt. This didn't show up, frankly, in the
6 reply or the reply declaration. So, the question is, has Petitioner met their
7 burden? Can they tell you this is what synchronizing means, here's our
8 expert's support, and here's how Coffman teaches it? They haven't even
9 gotten out of the gate. They can't even tell you what their expert believes
10 synchronizing means without -- as I said, here he spends four pages talking
11 about the things that you would need to evaluate to understand
12 synchronizing.

13 Slide 12 is just sort of a demonstrative showing these four pages of
14 deposition transcript testimony where he was talking, uninterrupted, about
15 how you would have to understand the term, synchronized.

16 On Slide 13, we point out, in the midst of this discussion, he says,
17 "So, synchronization may be required to be perfect." That's not what
18 Coffman has. And I think there's something -- I just want to point out, I
19 know that obviousness doesn't require verbatim disclosure, but Coffman
20 never says that a master device and a slave device synchronize their stacks,
21 it never says that. And in truth, Petitioner only spent -- if we were to
22 measure it, less than an entire page discussing updating. It's really the
23 bottom of page 30, the very top of page 31 of the petition. That's the limit of
24 their synchronization discussion. And there's no clear disclosure in Coffman
25 of synchronization.

26 There's no disclosure of why a master and slave have to have

1 synchronized context stacks. There's no discussion of even when a master
2 and slave switch operations that that requires synchronization of stacks, that
3 stacks have to be identical before you can swap. There's nothing like that on
4 the record. And, in fact, I would posit it to you that one of the reasons that a
5 master and slave device might switch is, the slave device may have a more
6 comprehensive context stack and may be better situated to serve as the
7 master device. And so, rather than synchronizing a stack, Coffman just says,
8 let's swap devices. You run the show for now.

9 Moving on to Slide 14, and this is a little bit relevant to Coffman.
10 Again, this is Mr. Schmandt's four-page testimony about what synchronizing
11 means. He says, "A proper understanding of what is required by
12 synchronization depends on what is the data and how carefully does the data
13 need to match between two different structures presumably at two different
14 computers?" He doesn't address these issues with respect to Coffman. He
15 doesn't ever go into Coffman and say, here's why a master and slave device
16 need to have the identical stack or synchronized stack.

17 JUDGE HOSKINS: Mr. Helge?

18 MR. HELGE: Yes, Your Honor.

19 JUDGE HOSKINS: Can you address the specification, which is,
20 of course, part of the intrinsic evidence that we have to consider for claim
21 construction --

22 MR. HELGE: Absolutely, Your Honor. Ooh, I'm sorry, Your
23 Honor.

24 JUDGE HOSKINS: I'm looking at Petitioner's Slide 34, which I
25 think has the passage in it that is the focus here.

26 MR. HELGE: And I think -- I know the one you're talking about,

1 Your Honor, and I'm happy you asked about that because we do need to
2 address that.

3 So, Your Honor, column 17, lines 15 to 32, I would agree, get very
4 heavily into the idea of synchronization in the '652 Patent. Here's the
5 important part from our perspective. When you look at the beginning of this
6 paragraph, it opens -- and I don't think this is on their slide, it opens,
7 "According to one embodiment of the invention, data and agents may be
8 stored and synchronized in mobile structures 10, mobile devices,
9 (INDISCERNIBLE) systems," so on and so forth.

10 So, it opens by talking about synchronizing data and agents. It
11 doesn't talk here about synchronizing stacks. I think it's talking, generally,
12 synchronizing context ==

13 JUDGE HOSKINS: Aren't stacks data?

14 MR. HELGE: Well, they include data, certainly. I think -- Your
15 Honor, I'm going to jump ahead a little bit. I'm not going to actually turn to
16 it, but I'm just going to tell you, Slide 19 of our presentation does address the
17 fact that both of the experts in this case agree that stacks have order and that
18 order is important. And this, we'll come back -- I'll discuss Exhibit 3002 as
19 well, Your Honor.

20 But if I can take a step back from our spec -- actually, before I do
21 that, I want to just finish. When you go through most of this paragraph,
22 Your Honor, at column 17, lines 15 to 32, most of this paragraph is dealing
23 with synchronization of data or agents. You don't really get to an entire
24 stack synchronization until the last sentence. And that shows up on the very
25 bottom left of Petitioner's Slide 34, and it's that sentence that begins, "In
26 some cases, a complete replacement of a database" -- they also say, "a

1 portion of a database." You can see Counsel has only highlighted a portion
2 of the database, "or of one or more agents may be performed rather than
3 performing a series of incremental changes."

4 Our position, Your Honor, is that the complete replacement of a
5 database is synchronization of a stack. It's the only instance where you have
6 a stack; i.e., a collection of data, or a database, in this disclosure, that
7 requires synchronization of, not just an individual piece of data, but the
8 stack. And that's what the claim is directed to, is the synchronization of
9 stacks. So, I think, Your Honor, I think that complete replacement of a
10 database is the exact support for what we're talking about in Claim 1, Claim
11 14, and Claim 20. We're talking about the synchronization of a stack.

12 Now, with regard to Exhibit 3002, Your Honor, I don't think this is
13 incorrect. I know Petitioner's Counsel talked about what we argued and
14 what we presented to the District Court, but I don't think the District Court
15 missed the point. And this is on page 22 of Exhibit 3002. It's about one,
16 two, three, four, five, six, seven lines down. The District Court says,
17 "Dialect's intrinsic evidence demonstrates that although a context stack need
18 not be an ordered list, it must be a data structure that can be ordered."

19 So, I think that, in combination with our Slide 19, which says,
20 "Both of the experts agreed that stacks have order." And then you have
21 Coffman, I think this is on Petitioner's Slide 14, order is important in
22 Coffman, order is very important. Because Coffman says, when you use an
23 application, that stack entry gets pushed to the top.

24 Now, if pushing something to the top is important in Coffman,
25 why isn't it important to synchronization? And so, here's -- I think, Your
26 Honor, we're all using computers a lot. Synchronization is something that's

1 sort of becoming -- into common vernacular among everybody, anybody
2 who uses a computer. You can have a calendar application on your phone.
3 You can have a calendar application on your computer. And if you add an
4 entry to your calendar on your computer, PTAB hearing, at 1:00 p.m. on
5 August 5th, and then you ask your calendar application to synchronize that
6 information, and that information shows up on your mobile application, but
7 it's at the wrong time, it's at 1:00 a.m. on August 6th or 1:00 a.m. on August
8 5th. You would say, those applications have not been synchronized, and
9 that's what's going on in a claim. We're talking about not just synchronizing
10 an entry, or synchronizing moving one application context higher. We're
11 talking about synchronizing the stack.

12 And this goes back to, Your Honor, our point regarding the
13 examiner. I don't think it's really necessary to get into why the application
14 was allowed to issue as the '652 Patent. But I think, one thing that happened
15 during examination that was helpful, was that the examiner did recognize
16 that synchronizing a stack is narrower. It's a narrower limitation than just
17 synchronizing information. And so, we would submit to Your Honor that
18 we would agree, that is a narrower thing, and that's not something that
19 Coffman discloses or suggests.

20 JUDGE HOSKINS: Mr. Helge?

21 MR. HELGE: Yes, Your Honor.

22 JUDGE HOSKINS: If we could return to Exhibit 3002 for a
23 moment?

24 MR. HELGE: Yes.

25 JUDGE HOSKINS: Didn't the Court there accept the argument
26 from Patent Owner that a context stack does not have to be ordered?

1 MR. HELGE: Well, Your Honor, two points about that. First of
2 all --

3 JUDGE HOSKINS: It's a pretty simple yes or no question. So,
4 you can make whatever argument you want, but I'm just trying to understand
5 the order. Didn't the District Court reject Patent Owner's argument that a
6 context stack -- excuse me -- reject Amazon's argument --

7 MR. HELGE: Hmm.

8 JUDGE HOSKINS: -- it was the accused infringer, that a context
9 stack has to be ordered?

10 MR. HELGE: I would agree with that, Your Honor. Yes, I would
11 agree with that. I think in the -- I'm going to say abstract, it's not really the
12 right word, but looking at that phrase alone, Your Honor, I would agree with
13 that.

14 Now, one point that is important is, in the '957 Patent, Claim 6,
15 there was also a requirement for synchronization, and in the '468 Patent,
16 Claim 15, there's also a requirement for synchronization. Those terms were
17 not construed -- it wasn't something that came up, I think, in that District
18 Court claim construction order.

19 But we would suggest also, Your Honor, I think that, again, given
20 that the experts in both cases have taken the position that order is an attribute
21 of a stack. Whether order is important or not maybe, just from the term
22 itself, maybe not important. But I think Coffman also shows us, as
23 Petitioner showed in Slide 14, in Coffman, order is important.

24 Whatever was accessed last gets pushed to the top, that means it's
25 important. It doesn't stay in place. It goes to the top. And so, if you're
26 going to synchronize and take something that's in Coffman, important at the

1 top, and not try to get something that's in the same place, I think you've
2 ignored something that Coffman thinks is important. And so, we would
3 submit that -- with respect, Your Honor, I understand what you're saying
4 about the District Court order. We think that synchronizing, as stacked here,
5 does require consideration of order.

6 And I don't think it's something that Petitioner's expert disputed.
7 We had this deposition. We talked about order. We made our response
8 arguments. He didn't come in, in his reply declaration and say, no, no, no,
9 no, order doesn't matter. Nobody cares about order. He didn't recant any of
10 that testimony. He didn't recant the position. It seems to me that he
11 accepted it.

12 JUDGE HOSKINS: Thank you.

13 MR. HELGE: And that's our Slide 19. Your Honor, I'm going
14 through my notes here. I think that I've covered all the points that I wanted
15 to cover, with the exception of one last point, and that is, once again, going
16 back to master and slave discussion that seems to be sort on the tip of
17 Petitioner's tongue today. And that is, again, looking at the petition, at the
18 bottom of page 30, at the top of page 31 of the petition, dealing with even
19 the concept of this master and slave existence, there is no theory of
20 inherency that they pointed to, to say that when you switch master and slave
21 you have to have synchronized stacks, or that you have to have identical
22 stacks, it's something that's not in that.

23 And then, the last point simply, Your Honor, in that same vein
24 -- I'm sorry, I thought I had a slide on this point, I may not have. But in the
25 same vein, Petitioner's theory that, if you only have one or two entries in a
26 stack, and you push them from master to slave, you could -- I think that the

1 argument was that you could have stacks that are synchronized, sort of an
2 even-if argument. And I would submit to you that the papers really don't lay
3 out the scenario in which you get synchronized stacks between a master and
4 slave device.

5 The petition certainly didn't. If the Petitioner had thought that was
6 an issue, it would have been front and center, and it's simply not in the
7 petition. So, I would say that this idea that, by chance there may be some
8 sort of synchronization when you were -- you only have one or two entries,
9 we don't even know the circumstances where you would only have one or
10 two entries, right? Even that scenario/hypothetical hasn't really been played
11 out by Petitioner.

12 Your Honor, for these reasons, I'll reserve the rest of my time, but I
13 think that Petitioners have failed to make their case on the independent
14 claims dealing with synchronization, and the dependent claims that we
15 discussed, dealing with the domain issues.

16 JUDGE HOSKINS: Thank you, Mr. Helge.

17 MR. HELGE: Thank you, Your Honor.

18 JUDGE HOSKINS: I don't have any further questions, and it
19 seems like my colleagues do not either. So, we have -- as you can see on the
20 wall, you still have five minutes from your principal case plus the five that
21 you reserved. So, when we come back to you, you'll have 10.

22 So, now, let me just pause for a minute and make sure that nobody
23 wants to -- or ask whether anybody wants to take a brief break here before
24 we turn to Petitioner's rebuttal?

25 MS. KEEFE: I wouldn't mind a three-minute break to go to the
26 restroom, but it's not necessary. If you all prefer to power through, I'm

1 happy to do so.

2 JUDGE HOSKINS: I don't want to make anybody uncomfortable
3 so let's just take a five-minute break, and we'll come back in five minutes.
4 Let's go off the record, and we'll come back in five minutes.

5 MS. KEEFE: Thank you, Your Honor.

6 (Whereupon, the above-entitled matter went off the record at 2:08
7 p.m. and resumed at 2:12 p.m.)

8 JUDGE HOSKINS: Back on the record. So, with that, Ms. Keefe,
9 you reserved fifteen minutes and then you had three minutes at the end of
10 your initial -- you've got 18 minutes. Use it as you will and begin when
11 you're ready.

12 MS. KEEFE: I do not intend to take all 18 of those minutes, Your
13 Honor, unless Your Honors want to engage in some kind of a dialogue,
14 which I'm absolutely happy to do. But I thought I'd start where we ended,
15 and then wrap back to the dependent claims.

16 I think what we heard was that all of a sudden synchronization still
17 has to be perfect, and it still has to be a swap of the complete database, but
18 that's simply not what column 17 contemplates. Column 17 of the patent is
19 exceptionally clear, the synchronization process. The only one that's
20 described anywhere in the specification. It's not like it's, oh, and then there's
21 another embodiment or something like that. "The synchronization process
22 attempts to determine which version of a data element, or an agent, are the
23 newest or most up-to-date and propagates that element." And so, it's talking
24 about finding, "the newest thing."

25 And in Coffman, what we see, and we actually heard from Patent
26 Owner's Counsel is, Coffman is saying, hey, push up the most recent

1 application that we used to the top. That will now be the newest. That's
2 what we need to update.

3 And so, even if he wants to say, but it has to be the one on the top,
4 Coffman says, yes, it's the one on the top, let's grab that, let's pass that
5 information to coordinate our conversations. And so, the definition that we
6 are using matches up to Coffman.

7 And then, alternatively, what we had been trying to point out was,
8 even if synchronization were required to be perfect, which we know that the
9 case law says it does not have to be, there is no requirement from any
10 embodiment of that, in our reply at pages 7 through 8, and this is in our Slide
11 38, which I haven't shown you yet, we talk about the fact that occasionally,
12 in Coffman, there is a discussion or a description of having the role of
13 master and slave be dynamically switched. And the only way that it would
14 work for them to be dynamically switched, in other words, the slave
15 becomes the master, is if everything else within those stacks were otherwise
16 identical except perhaps, for the very last thing, which would then have to be
17 updated.

18 And our expert discusses that in his expert reply declaration at
19 paragraph 11. By having that identical content and ordering, the master
20 dialog manager would better be able to carry out its responsibilities,
21 including helping to make sure that decisions that are based on what's at the
22 top of the stack is using the right information. So, even under that
23 circumstance, there would be synchronization. But again, the patent simply
24 does not require that level of perfection, and their own argument to the
25 District Court was that synchronization in that stack -- sorry, not
26 synchronization, that the stack itself did not have to include a specific order.

1 I would then turn Your Honors, very briefly, to a discussion about
2 the dependent claims. And I'd like to push first for Your Honors to look
3 back at their Slide 23, which talks about what they call Coffman's task
4 dispatcher. And it's very important what language we jumped over when we
5 were reading that language from Coffman. Coffman says, specifically, that
6 the "Task dispatcher/controller feeds input from the conventional and
7 conversational subsystem services to the context stack, "-- here's the key,
8 "via the dialog controller."

9 Now, if we look back at our Slide 10, which shows how Figure 4 is
10 represented, the dialog controller is talking to the context stack. That's on
11 the back-end. The only place that we mapped Figure 6 was for the front-end
12 where a microphone passes information into the system, which is then
13 parsed to recognize the speech, turn it into text, which will then further be
14 processed.

15 And so, in fact, the dispatcher here, has to be working via the
16 dialog controller, which is working with the context stack in order to grab
17 the applications that are going to be used. We kept hearing that it's not about
18 the applications, but that's exactly what Coffman is about. Coffman is about
19 trying to find, how am I going to parse a request for spreadsheet information
20 when I don't even have a spreadsheet open? I'm going to parse down, or
21 track down the context stack in order to try to find the application that I
22 want. And then, as Claim 11 tells us, you're going to select the appropriate
23 application.

24 We also see that exact thing happening, even in the alternative
25 embodiment, where a single application had its own CVM. If there were no
26 matches made in that stack, then Coffman would go directly to the next

1 stack and the next device in order to try to find a match.

2 And so, again, it's the CVM that is trying to select the application
3 that is going to be used. So, I hope that --

4 JUDGE HOSKINS: Ms. Keefe, I --

5 MS. KEEFE: Yes.

6 JUDGE HOSKINS: -- this is Judge Hoskins. I think Patent
7 Owner's -- my response to that might be, and I'm looking at your Slide 10.
8 We have underneath, if you will, below in these figures, the CVM, we have
9 conventional OS and API, and then hardware at the left side of your slide,
10 and then you've cut it off. But in the right side of your slide, where you've
11 got part of Figure 4, there's that little up arrow going into the box 404, which
12 had some similar components.

13 And I think, Patent Owner may say, well -- when it talks about
14 choosing applications, it's talking about choosing those applications that are
15 below the CVM rather than choosing either the conversationally-aware
16 applications 11 or the conventional applications 12. So, I'm wondering if
17 you can address that?

18 MS. KEEFE: Absolutely, Your Honor. And I will go back again
19 to the example given in the patent -- sorry, I keep calling it a patent, in the
20 Coffman prior art as well. And Coffman, as we've discussed, talks about
21 managing Windows applications throughout. The whole point is to try to
22 figure out which application we're going to pick up. Subsystems, which are
23 what are at the bottom there, are not referred to as applications. So, we have
24 the subsystems over on the left-hand side of Figure 1, the conversational
25 subsystems. Those are not referred to anywhere in Coffman as applications.

26 Instead, applications are things like the spreadsheet, like Excel, or

1 the email program. And the example given in Coffman was, while someone
2 is checking their email, they can ask for information from a spreadsheet.
3 There's no disagreement that the spreadsheet application then has to be
4 found, has to be activated, has to then be searched to find the information so
5 that it can be returned back to the user.

6 So, I don't understand how the information, passing in, can be
7 anything other than the conversationally-aware applications. Those are what
8 are described in the Coffman prior art as the Excel spreadsheet -- I don't
9 think they use the word Excel, they just call it a spreadsheet -- and the email
10 program. And so, those are the applications that are being selected, and
11 that's exactly what Claim 11 calls for.

12 I would also refer Your Honors back to the notion that they seem
13 to be mapping an embodiment where each application has its own CVM, but
14 that's not what we mapped. What we mapped was -- when you look at Slide
15 16, which is Figure 10, each of those boxes could be either a device or an
16 application. And if it was a device with its own context stack, that stack
17 would include the applications that are being invoked, including the notion
18 of popping the most recent one back up to the top that everyone seems to
19 agree is important to Coffman.

20 I hope that answers Your Honors questions. Unless you have any
21 others, I think I'm finished.

22 JUDGE HOSKINS: Thank you.

23 MS. KEEFE: Someone once told me I would never be punished
24 for taking less time not more.

25 JUDGE HOSKINS: I would agree with that philosophy. Let me
26 just pause for -- it looks like my colleagues also do not have any other

1 questions. So, thank you. And so, then we'll go back to Mr. Helge. So, Mr.
2 Helge, I'm looking at the clock -- okay, so you have 10 minutes for your
3 rebuttal, and you may proceed when you're ready. Thank you.

4 MR. HELGE: Thank you, Your Honor. And like Ms. Keefe, I do
5 not intend to use it all.

6 Your Honor, two very quick points about Petitioner's rebuttal.
7 Regarding Slide 38 in this discussion of the master and slave devices, and
8 the swapping of stacks, what I heard Counsel say was that the only way that
9 this would work is if somehow the stacks were synchronized. And they
10 have not made a case of inherency. Just to be clear, there is no inherency
11 argument that just because the master and slave can switch devices, that that
12 requires some synchronization. That argument has never made. The proof
13 has not been set forth. We would disagree with it.

14 Again, as I think I mentioned to Your Honors earlier during my
15 presentation, one reason to swap master and slave is because the slave has a
16 more robust context stack and it's better equipped to handle a new request or
17 may have more information regarding an application, for example. So, I just
18 want to make clear that there's no inherency argument that should be
19 considered by the Board, they haven't met the case, and that should not carry
20 weight.

21 Lastly, Your Honors, regarding our Slide 23 and their Slide 10, I'm
22 looking at those two against each other as Counsel was talking about them.
23 There was some argument made about that we had skipped over certain
24 relevant in Slide 23. Now, we don't dispute that the dialog controller selects
25 the active context, but the active context is not selecting a conversationally-
26 aware application. Those are different concepts. We don't agree that

1 selecting an active context -- or trying to figure out when a command,
2 request, or query comes in from a user, that the dialog manager go into the
3 stack and figuring out what's active is also figuring out which
4 conversationally-aware application to send a command back to.

5 Now, Judge Hoskins, as you pointed out regarding their Slide 10,
6 there is some information cut off above the task dispatcher/controller. But
7 when I'm looking at this disclosure from Coffman, page 31, lines 15 to 18,
8 from our Slide 23, and looking at their Slide 10, Coffman says that the task
9 dispatcher/controller sends output of the different tasks -- it says to the
10 different subsystems and prioritizes them.

11 And as we can see, the conversational subsystem services are 406
12 on their Slide 10, which is also Coffman's Figure 4. It doesn't say it's going
13 back up to a conversationally-aware application to process something, and
14 then come back with an answer, and then send the answer back to the user,
15 which is I think what Petitioner's theory is what's going on here.

16 And actually, if you were to look at Coffman's Figure 4 to the right
17 of the conversational subsystem services 406, what you have is another
18 block 407 with speech recognition, speaker recognition, the NLU, the TTS,
19 but you don't have conversationally-aware applications available in that
20 block to try to process a user command or request.

21 Lastly, about their Slide 10, and I just want to point this out to
22 Your Honors, they're citing petition at page 25. They did not rely on
23 Coffman's Figure 4, or its embodiments, to make their case for dependent
24 Claim 8, Claim 18, 24, and those other dependents, they relied on Coffman's
25 Figure 6. And if you recall, we saw in Coffman's Figure 6, the back-end
26 applications 608 are frankly on the opposite side of the CVM from the user

1 and from the conversationally-aware applications that the user would be
2 accessing. So, we don't think that Slide 10, or this Figure 4 that they've
3 referred to, answers the question. The task dispatcher does not go back to a
4 conversationally-aware application to try to figure out how to process a user
5 command, request, or query. Coffman does not say that.

6 Lastly, again, Your Honor, one more point about the spreadsheet,
7 and the email, or whatever they're talking about in, like, page 56 of
8 Coffman. Coffman doesn't tell you how those applications being accessed
9 by the user are then interacted with a context stack. Simply, it's very
10 general. It doesn't give you the detail that you need to know, whether Claim
11 8, 18, 24 are being practiced by Coffman. Frankly, we would submit, based
12 on everything that Coffman says, that's not how Coffman works.

13 If Your Honors have no other questions, I thank you very much for
14 your time today in the discussion, and I will submit our case.

15 JUDGE HOSKINS: Okay. It looks like we do not have further
16 questions. So, I want to thank both counsel for really an excellently
17 presented oral argument today. The case is submitted, and you can be
18 expecting a final written decision in due course.

19 (Whereupon, the above-entitled matter went off the record at 2:27
20 p.m.)

IPR2024-00752
Patent 8,849,652 B2

PETITIONER:

Heidi Keefe
Andrew Mace
Reuben Chen
Mark Weinstein
COOLEY LLP
hkeefe@cooley.com
amace@cooley.com
rchen@cooley.com
mweinstein@cooley.com

PATENT OWNER:

Wayne Helge
James Wilson
BUNSOW DE MORY, LLP
whelge@bdiplaw.com
jwilson@bdiplaw.com